Issue 16 | May 2008

Blender learning made easy

blender art MAGAZINE

Wow Factor

Skin Shading using multi-layered SSS

Realistic Smoke in Blender

Burn Them All

Making hair for Wolf

COVERART - 'Hans Schwaiger - Mice'

blender art MAGAZINE

EDITOR
Gaurav Nawani gaurav@blenderart.org

MANAGING EDITOR
Sandra Gilbert sandra@blenderart.org

WEBSITE
Nam Pham nam@blenderart.org

DESIGNER
Gaurav, Sandra, Alex

PROOFER
Kevin Braun
Phillip Ryals
Bruce Westfall
Joshua Leung
Lynda Schemansky
Eric Pranausk
Noah Summers
Joshua Scotton
Mark Warren
Wade Bick
Patrick O'Donnell
Brian C. Treacy
Scott Hill
Henriel Veldtmann

WRITERS
Slavoljub Pantelic
Victor Malherbe
Georges Mignot
Daniel Salazar
Jean-Sébastien Guillemette
François Grassard
Jack Harris

COVER ART
Hans Schwaiger - Mice

CONTENTS

Sandra Gilbert
Managing Editor

In a digital world where most everything is faked or is a work around of some kind, the desire for ever more realistic effects and models, faked or not, has CG artists always wishing for more. Especially when it comes to hair/fur, water simulation and a realistic cloth modifier, just to name a few.

And while a great many great animations and images never make use of high end techniques, tools and effects, at some point all CG artists wish to play with these concepts and tools, if for no other reason than "it's cool!"

CG artists do understand that not every feature wish is possible, but luckily for us the blender coders are very talented at giving us as many of these wished-for features.

Yay, us!

The Peach project (Big Buck Bunny) gave us some very nice improvements to the particle system allowing for some truly beautiful hair/fur. When set up right, it looks so soft, you just want to reach out and touch it.

After years of development, our long wished for cloth modifier has arrived. The new cloth modifier will herald a new age of test images filled with nicely draped clothes, blowing flags and all manner of fluttering cloth.

And who can resist setting up a massive flood or tidal wave scene with water crashing everywhere. Or better yet, create a beautiful work of Fluid art.

In this issue we cover how to use some of these great tools as well as how to set up some very cool effects. So go find your favorite chair and settle in for a great read.

Happy Blending!

sandra@blenderart.org

IZZY SPEAKS : DVD's, New Releases, Oh My!

All the animals looked so soft and furry that I Just wanted to reach out and squish them.

Big Buck Bunny:

Oh My! It arrived so fast. I hadn't expected my copy to show up for a few more weeks. I had other things I planned for that day, but it wasn't too long before I was watching BBB. I just couldn't wait!

I'm not going to spoil the story line for anyone who hasn't seen it yet, but I will say that I found it to be very funny. The characters had great personalities and it was great fun to watch their expressions change as well as how they moved and interacted with each other.

The look and style of BBB was just beautiful. All the animals looked so soft and furry that I just wanted to reach out and squish them. The colors of the environment were soft and dreamy. You can be sure that I will be digging through the production files in order to find all the texturing and material secrets.

In fact, I will be studying the production files (as I'm sure everyone will be) quite thoroughly to see how they got not only the great materials and fur, but the lighting as well.

I haven't yet had time to look through all the files and extras on the DVD's, but a quick look assured me that there is a lot to see and play with. If you haven't ordered your copy yet, I encourage you to do so as a soon as possible. You won't regret it.

New toys to play with:

Although I keep up with Blender's progress and development, with my schedule I don't often have time to play with test builds. Which is something of a bummer considering the sheer number of new and/or improved features that seem to get added almost daily. To make matters worse, even when I get my hands on the latest official release, I still don't have time to really sit down and just play with new features. Like many of us, I learn and explore features as I need them for projects. And lately my projects haven't really needed any cool features to work. Double bummer (note to self: think of better projects so you can play with new features)!

The arrival and release of Blender 2.46 didn't increase my free time, but the fun new tools were too tempting to resist. So after a quick browse through my old files for a suitable model to experiment on, I sat down for a bit of 'blending' fun. Shhhh! Don't tell anyone, they all think I'm actually working on something productive.

Here are some of my impressions. I don't often just sit and push buttons to see what they do, so this might take a while.

Some time later . . . okay, 2-3 hours later (I was having fun!).

The Cloth Modifier: Even though I am not a fashion designer, I had great time playing with a very basic poncho (a simple circle, subdivided and shaped just a little). It wasn't a fancy test by any means, but it was great fun to play with the defaults and watch it drop and fold around my little character. I was able to get nice results with little to no tweaking; just using the supplied presets created a nicely draped poncho.

It was interesting to see how the different presets acted and varied from one another. My poncho was fun, but even I am entertained by a falling poncho for only so long. So I moved up to an ugly robe-type outfit and added some wind. That entertained me for quite a while. I was able to view the robe fluttering in the wind from any angle (yes, I am very easily entertained). Silk was my favorite to watch. It has a nice sliding quality to it, that let the cloth just flow over the character. The rubber was quite entertaining when wind was added, as it bounced off my character's belly and flapped around a bit.

IZZY SPEAKS : DVD's, New Releases, Oh My!

There are options for custom cloth, but honestly I think most needs would be adequately met with the presets. Especially if you are just messing around as I was.

Fur/Hair (Particle system): On to hair and fur. Since most of my characters are animals of some sort, I was looking forward to playing with the newly improved particle system. A quick search for instructions led me to a WIP tutorial. It gave some nice tips on getting a fur system going; then I just played from there. My model ended up looking like a cousin to Big Foot, but he was definitely furry. Great fun! The hair editing brushes kept me busy for some time as I kept restyling his hair. I obviously need to take a few hair styling classes . . . my little guy was surely having a bad hair day.

I had actually meant to play with and explore more of the new features, but I spent so much time entertaining myself with watching cloth fall and styling hair, that all too soon it was time to get back to all those other things I was supposed to be doing.

There were other features I really wanted to try out: UV Editing, Render Baking, Bone Heat weighting and the Armature Drawing improvements . . . but I'll have to return to these tests later and play some more when I have more time, or when no one is looking.

ManCandy FAQs:

Over the last several months (okay, probably longer than that) I have been trying to learn about rigging and animation. I would really like to expand my animation skills from moving of simple objects to character animation, so I was really excited about the ManCandy FAQ DVD. I saved up my pennies and finally ordered it. After waiting for the usual 2-3 week delivery time, (it would appear that I live on the ends of the earth when it comes to deliveries) it finally arrived. Yay!!!

I was so excited, even though I really had way too many other things I needed to be doing, I couldn't resist popping the DVD into my computer to take a sneak peak. Big mistake, even just poking around the DVD to check things out left me wanting to sit down and devour the entire thing. But, I really, really had other things I needed to be doing. Bummer! Big bummer!

Sigh . . . so I proceeded to take care of all the things that needed to be done. One week passed, and then two. Still no time to watch my new DVD. This really sucks. At this rate it will never get watched.. Then all of a sudden I had an amazing idea. I grabbed my DVD, my old laptop and took them both to my craft room, where I promptly set it up and started the DVD. I have been listening to the DVD while working on my projects. Problem semi-solved.

I still haven't watched very much of the DVD (just bits and pieces here and there), but I have listened to the whole thing numerous times. Like I said, not a perfect solution, but I am learning quite a bit. Granted, I still have to actually watch it all, but this will do for now.

I have long admired Bassam's skill and creative vision, and after watching the DVD, I am pleased to discover that he is also quite the teacher and rather funny too. His teaching style is akin to having a long in-depth conversation with a friend. He explains everything in an easy going, casual manner. One thing I have really enjoyed is how he not only shows you how to set things up properly and why, but what happens if you set it up wrong or forget to set something up. He even left some mistakes of his own in the videos and then showed how to correct them.

Bassam packed an amazing amount of information into each video. And while they obviously all go together, they also stand alone as full lessons within themselves. I have already listened to the DVD many times and will listen to and watch it many more. Every time I "watch" a segment, I learn something new and my understanding of rigging increases, which can be only a good thing.

The ManCandy FAQ DVD was well worth the money and the wait. Now, if only I can just find the time to actually do something with all my new-found knowledge.

Blender 2.46

The work of the past half year - and the efforts of the open movie project "Big Buck Bunny" - have resulted in a greatly improved feature set, now released as Blender 2.46, the "Bunny release"!

This version supports a new particle system with hair and fur combing tools, fast and optimal fur rendering, a mesh deformation system for advanced character rigging, cloth simulation, fast Ambient Occlusion, a new Image browser, and that's just the beginning. Check out the extensive list of features.

Videos of 2.46 Functionality

A Blender artist from the French Blender forums Blender Clan, has published a set of videos, showing how to use some of the new features of Blender 2.46. There are over twenty videos available to download!

Here is a short list of the videos available:

- Explode modifier demo.
- Meshdeform modifier demo.
- Some of new composite nodes in demo.
- Demo of render stamp function.
- Demo of the new hair system (particles), with traven as an outstanding hairdresser.
- Cloth function demo (not very original, but at least it exists).
- changes about the new UV Mapping system (edit mode / UV mode have been merged)
- soft shadow demo with a ray spot
- bone heat weighting and Pose lib demo (and some other things)

Be sure to check out the full list of available downloads, there is a lot to learn and see.

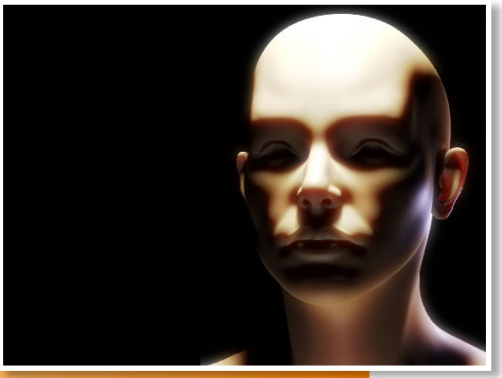
Project Apricot Packagers needed !

We are preparing to open our repository for the public to feel real Open Source power. At this moment we need people who will help us with making package releases for GNU/Linux, Windows and OS/X. So, if you have time, if you know something about packaging and bug tracking/solving write to us ! Apply to: res[-a-t-]crystalspace3d.org

Project Peach: BIG BUCK BUNNY

All Pre-Ordered DVD's have been shipped and are making their way around the world, you can track the arrival dates on the frapper map.

Online release of BBB along with all production files went smashingly well. You can find a list of all mirrors and torrents at the BBB site.



By Victor Malherbe

Introduction

Organic materials are probably some of the most difficult to setup, since their realism does not rely solely on the quality of their textures. CG skin often looks plastic, and, when subsurface scattering is turned on, skin tends to look like wax. Over the past few weeks, I have been trying to achieve realism when rendering skin, or at least, to obtain a skin shader which would not look terrible when rendered.

After some research, I began to build a kind of "multi-layered" skin shader, which could take into account the various layers of human skin. In Blender, it means using one material for each of the layers of the skin (epidermis, dermis, and so on), mixing them together within the material nodes editor so as to obtain the final output.

PART ONE : LAYERING THE DIFFUSION

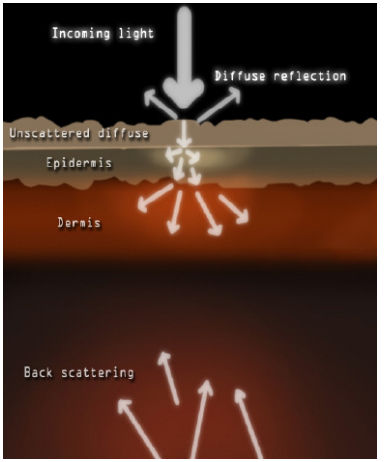
Step One: The Structure of Skin

First, let's deal with the diffusion of our material: we will setup the specular highlights separately. Before trying to simulate the appearance of skin, we must understand its composition:

- The upper layer of the skin is the epidermis, which scatters the light slightly. It is yellowish/grey. Under the feet for example, the epidermis is quite thick, which is why the skin appears almost yellow.
- Then comes the dermis - it is almost red, because of the blood. Unlike the epidermis, it scatters the light a lot.
- Though the underlying tissues (muscles, organs, cartilage, etc.) are almost unnoticeable, we will have to simulate a kind of "back scattering" - when the ears are lit from behind, for instance, they appear red. The back scattering could have been done within the dermal layer, since there is a "Back" factor in the SSS panel of Blender, which allows setting the front scattering and the back scattering separately. But in the end, it enables more control to separate the back scattering from the dermal scattering.
- One last note about the structure of skin: in theory, there is nothing above the epidermis. However, using SSS (even in a very subtle way) will subdue the bumps of your textured models. That's why we will have to apply another layer, which will be an un-scattering diffuse layer. Doing so, you will be able to conserve all the bumpiness of your textures, and only this layer will be textured (concerning the diffuse shaders).
- Simply take a look at the drawing, which sums up what we need to simulate. Knowing all this, we are able to begin Blending, and setting our materials.

Subsurface Scattering

Subsurface scattering, also known as SSS, is a phenomenon which has to do with the translucency of some objects, which is to say that they are not fully reflective, and neither transparent. The light beams simply bounce on reflective surfaces. They go through transparent and uniform objects, being deflected two times (when they hit the surface, and when they exit the object). But translucent objects scatter the incoming light, deflecting the beams several times in a chaotic way before the beams exit. This produces a very soft looking surface. Objects that exhibit this behavior are, for example, milk, wax, potatoes, etc. Many organic objects tend to scatter light.



Step Two : The Un-scattered Diffuse

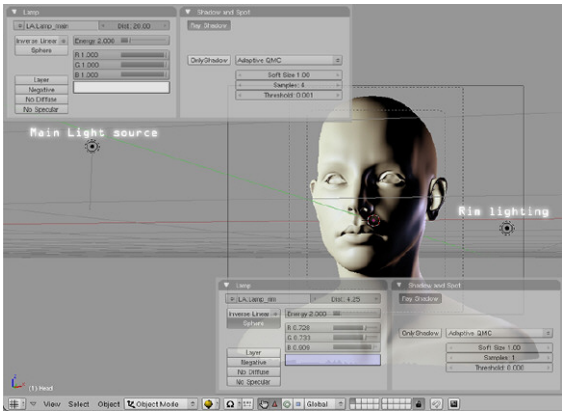
Lighting setup: before starting the shading, we have to set up a quick, and yet usable, test scene. I've used a head model provided by Maqs on the BlenderArtists.org forums, who also gave me permission to display his model in this article.

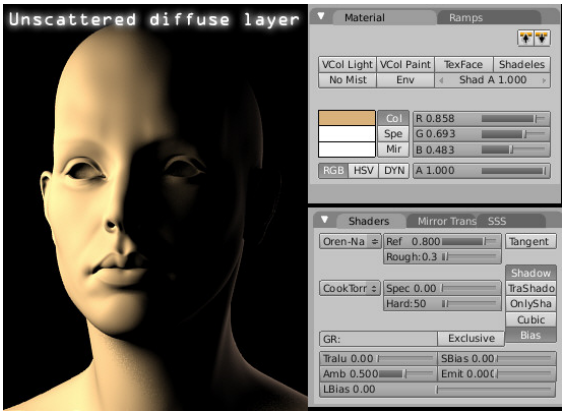
He deserves all thanks, since he also helped me a lot in building my shader. You can also use the default Monkey head mesh instead of Maqs' head model ([Space] Add»Mesh»Monkey), or even a head imported from MakeHuman (File»Import»Wavefront).

My lighting setup is quite simple: a main Lamp (Add»Lamp»Lamp) to illuminate the head (with soft raytraced shadows turned on) and a "Rim" lamp with a slightly blueish tint, lighting the object from behind in order to make the edges stand out.

Now we can start the shading itself. Add a new material to your object (in the Materials panel, «add new»), and call it "un-scattered_diffuse". Give it a natural skin color, and set the diffuse reflection to Oren-Nayar. The Oren-Nayar diffuse shader takes into account, when rendering, hypothetical microscopic bumps on the surface, which allows it to simulate rough surfaces such as clay, clothes, dry rocks, etc.

So it helps achieve a more natural shading. Don't forget to put the Spec slider to 0.00, since we will take care of the speculars later. If you painted some nice textures for your model, apply them now, except for the specular textures. It is important to understand that even though the epidermis is the upper layer of the skin, we will not texture it, since the scattering effect would blur the bumps too much.



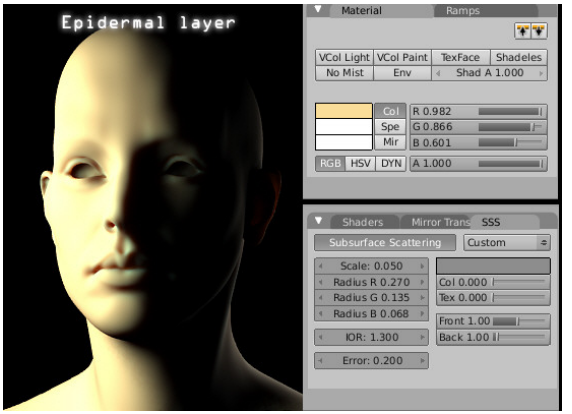


by Victor Malberbe

by Victor Malherbe

Step Three : The Epidermal Layer

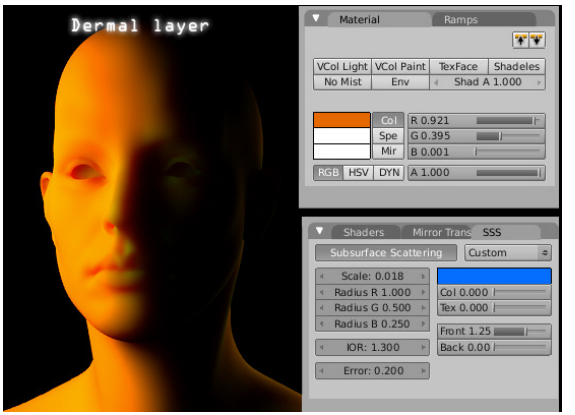
Now it's on to the epidermal layer. We have to build a shader which should look like a corpse without blood, since the blood flows mainly through the dermis. In the end, it has to look gray and yellow, and needs a quite subtle SSS effect. After having added another material called “epidermis”, give it a yellowish gray color as if it was desaturated skin, and take a look at the settings I've used for the SSS (I will not detail everything, since we have to set three layers using SSS; I prefer using screenshots). Just keep in mind that for every layer, the Red radius will be twice as large as the Green radius, and the Blue radius will remain two times smaller than the Green radius (based on scientific measurements)



Just a quick note about the SSS scale: in theory, we should be using exactly the same scale factor for all layers incorporating SSS, because our object has a fixed size. However, since the SSS radii of epidermis, dermis, and back scattering differ strongly, it would lead to handling tiny SSS radii for the epidermis, and huge SSS radii for the other layers. Operating in this way would be imprecise, and because we're working step-by-step it is actually better to set each layer separately, without wondering if the scale factor is the same for every layer.

Step Four : The Dermal Layer

In reality, the dermis is red or orange, because of the blood flowing through it. When it is time to combine all diffuse layers together, we will need to be able to notice the underlying blood of the skin. So, add another material and name it “dermis”, and set a rather reddish orange tint for it. Unlike the epidermal, the dermal layer has to scatter light a lot, so we are dealing with far bigger SSS radii. Simply check the screenshot so as to see the settings I've used. Note that we put the “Back” factor to 0.00, since the back scattering will be handled with another shader. Once again, don't forget to put the Spec slider to zero.

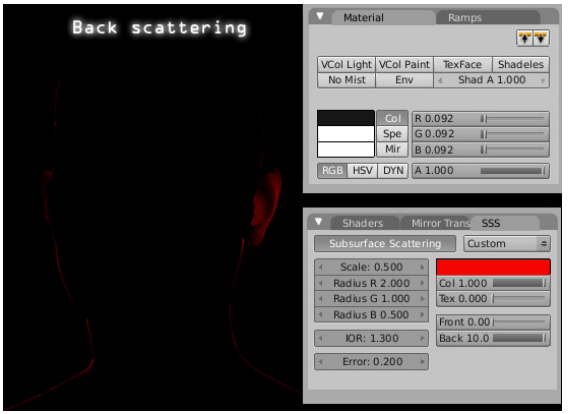


Note The SSS color is set to blue. Indeed, I have noticed that setting the SSS color to blue caused the terminators (transitional areas between light and shadow) to take on the complementary orange tint, and that is what we want to happen.

by Victor Malherbe

Step Five : The Back Scattering

Now we can take care of the back scattering. I have preferred to separate it from the dermal layer. Doing so, we have more control on the subsurface scattering effect. Add yet another material called “back_scattering” for example, and set its color to almost black, but not pure black - if your shader or texture has a pure black area, nothing will ever scatter because any color multiplied by black is still black (pure black is coded by [0 , 0 , 0]). Now in the SSS panel, turn the SSS color to red, and set the SSS radii to be quite large. It actually depends on the size of your model, but in my case I wanted to see some red on the ears, but not too much (otherwise, it would have given the feeling that the model was really tiny).



Note that the “Back” factor of the SSS is at its maximum, whereas the “Front” factor is turned to 0. Again, put the Spec to 0.

Step Six: Mixing All Diffuse Layers Using Material Nodes

Let's start: add another material called “Combined_shaders”, and turn on the Node button. Open the Material Nodes editor, and within the editor load each material we created previously, using [Space]›Add›Input›Material. Do it four times, and for each node assign one of the skin materials. Now it's time to mix all of the layers together. Add a screen operator using [Space]›Add›Color›Mix and change the mixing method to "Screen" instead of "Mix". Now, link the epidermis to the first input of the "Screen" operator, and link the dermis to the second input. Leave the mix factor at the default value, which is 0.50. Now you can do the same with the “Back_scattering” material, using an “Add” operator this time.

The back scattering will be black, except in areas where the model is lit from behind. Using "Add", we are assured that the dark areas of the back scattering won't change anything in the render, since adding pure black ([0 , 0 , 0]) changes nothing. Take care to set the “Fac” to 1.00, otherwise the back scattering will not be fully added. Of course, if you notice that the back scattering it too strong, you can easily reduce the “Fac” instead of changing the original SSS parameters. And finally, combine the “unscattered_diffuse” with the other materials using another "Screen" operator.

This time, I turned the “Fac” to 0.60. You can decide if you want to see a rather translucent skin, or a quite reflective skin which does not absorb a lot of light. To finish, link the "Screen" node to the “Output” material.

Note

“Error” parameter: for tests renders, you should set it to about 0.5, or even 1.0. For final renders you can set it to 0.1, or even 0.05 if you want, in the case of a large render and/or more accurate calculation. In my quick tests I landed at 0.2, since I wanted quite clean renders without having to wait too long.

That's all for the diffuse shaders. Now we have to mix all of these together within the Material Nodes editor in order to render all layers at the same time.

by Victor Malherbe

You may have noticed that I inserted an “RGB Curves” controller after the combination of the Epidermis and Dermis. Indeed, it is far better to tweak the color of the skin this way, rather than having to change the SSS parameters in the materials panel.

One last thing about the "Screen" operator: in theory, we should be using a “Mix” operator to combine the layers, since we simply want the various layers to appear all together. However, the "Screen" operator provides better results, whereas a "Mix" operator tends to make the skin look flat. You should also use a "Screen" operator, in my opinion, provided your lighting setup is not too “extreme”, so to say. In extreme lighting conditions, for example, a very strong sunlight, using a "Screen" operator will make the colors totally wrong. In the end though, whether you use "Mix" or "Screen" actually depends on the specific situation.

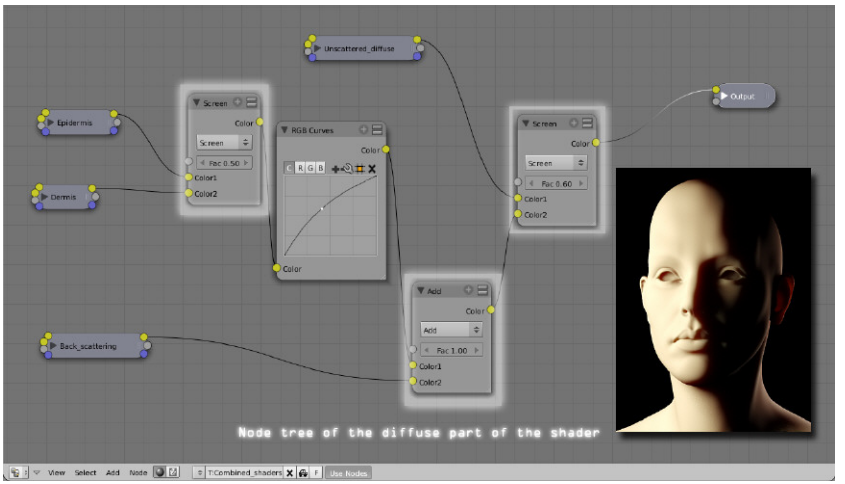
It is starting to look pretty good. And yet, it still looks like rubber. Seemingly, some specular highlights are missing . so let's jump to part two!

PART TWO : ADDING HIGHLIGHTS

Step Seven: Specular Reflections on Two Layers

The highlights that we intend to simulate come from two sources: the skin speculars (which are quite soft and large) and the speculars due to the sweat on the skin, which will be brighter and more concentrated. Both specular terms will have a blueish tint, so as to counterbalance the yellow color of the skin diffusion. I personally chose to use the default Cook-Torrance specular shader, since it allows us to simulate a kind of Fresnel effect, unlike the Phong specular shader, for example (highlights have to appear brighter at grazing angles)

Add another shader to your model and name it “Skin_soft_speculars” for example. Don't forget to put the "Ref" value to 0.0, since we are not dealing with the diffusion anymore. You can also set the base color of the object to be pure black.



Note

When talking about reflective surfaces, the Fresnel effect is often involved. Indeed, the reflectance of most surfaces is not the same at grazing angles, compared to the reflectance of these surfaces when seen head-on. This is called the Fresnel effect.

You can see an example of this phenomenon if you look at a window: at grazing angles, you cannot see through it, the surface of the glass being more reflective than when viewed normally.

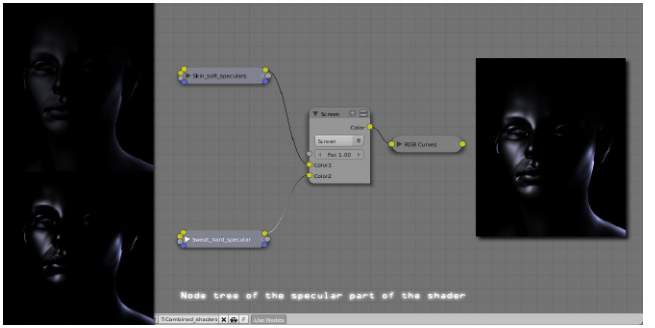
Or in the node editor, when you have added your materials, you can turn off the “Diff” button. The most important is that these shaders only provide speculars. The Cook-Torrance specular shader is activated. Since we want the skin speculars to be soft and large, set the "Spec" value to 0.1, and the "Hard" value to 10.

After having put the specular color to a slightly blueish tint, you can go on and create another material, which will be called “Sweat_hard_speculars” or something similar. The only difference is that the "Spec" slider is set to 0.3, and the "Hard" value to 30.

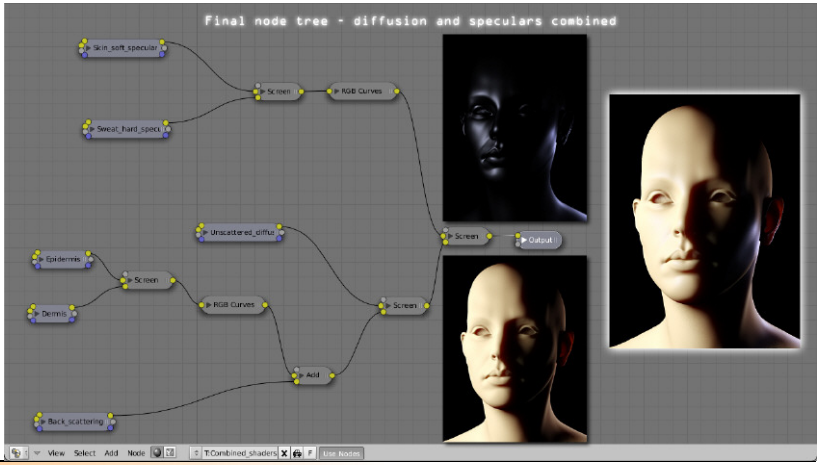
Step Eight: Mixing the Specular Terms with the Diffusion — Final Nodal Tree

Both layers of speculars are now done. Select your “Combined_shaders” material, and open the node editor once again. Start by adding your specular materials in the node setup. Mix both specular shaders with another “Screen” operator (as usual, [Space]»Add»Color»Mix and turn the mix method to "Screen"). Set the Screen factor (“Fac”) to 1.0. After this screen operator, you can insert another “RGB Curves” controller, as I did, since it enables faster control for the brightness of the speculars.

To finish, add the last "Screen" operator, which will screen the speculars on top of the diffuse part of the shader. Set its factor to 1.0. The image above shows the nodal tree concerning the specular part of the shader, and the screenshot below shows the completed nodal network.



Our shader is now ready to use. Of course, there are lots of aspects that could, and should be improved in order to achieve photo-realism. If you are not yet growing bored, perhaps you would like to go on reading. Part Three awaits you!



by Victor Malherbe

by Victor Malherbe

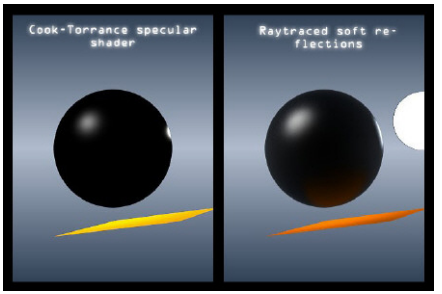
PART THREE : GOING FURTHER

In this part, I will try to give you some tricks/tips, an overview of what could be improved in the shader we've built, and beyond that some rendering tips aimed at achieving more realism.

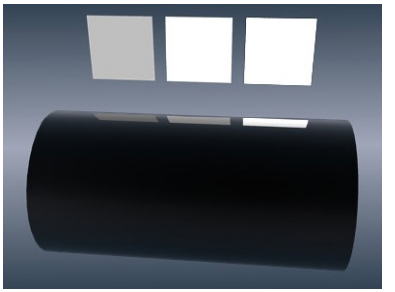
1. Proper Highlights Using Soft Raytraced Reflections

You may have noticed several times that specular highlights are a poor way of simulating the real behavior of light. In theory, highlights occur because brighter-than-usual objects are reflected. Speculars are in fact a cheap way of obtaining highlights. They are not able to reflect anything else that lights objects, and on top of that (in Blender at least) specular highlights do not match the correct shape of the reflected object.

Simply take a look at the comparison on the right, and you will notice that first, specular highlights do not reflect the environment (the orange plane and the blue sky). Then, you should see how strange the shape of specular highlights become at grazing angles. They look round, whereas they should have the shape of an arc of circle, just like on the render with soft reflections.



What if we tried to do correct highlights using raytraced reflections? It would be expensive in terms of render-time, of course. Yet, at least we will obtain something more satisfying. Let's give it a try. All we will need to do is add another shader, which will use only soft reflections, and then combine it with the original shader. I will leave it up to you to decide whether you want it to fully or only partly replace the speculars, due to the amount of render-time.



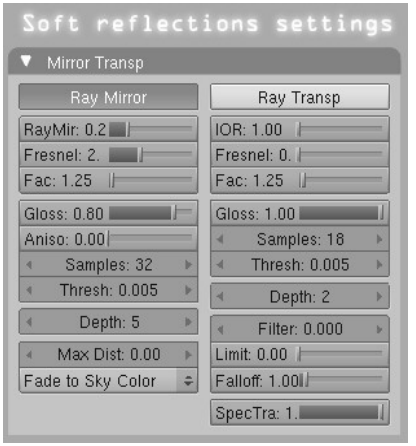
Our first consideration is that since we want to see highlights only, our surface cannot be fully reflective. So how are we going to see white highlights if our material only reflects 50% of the light? White objects would appear only 50% white, when reflected at 50%. The answer is simple: thanks to the “Emit” parameter, we are able to shade objects that are "whiter than white”. Doing so, we will get proper highlights on a surface which does not reflect all light. Just check the render below, which shows the reflections of three planes. These three planes only have one difference: their “Emit” value is increasing, from left to right.

If you want to use soft reflections instead of specular highlights, or at least combine them together, you should keep in mind that reflections do not show the Lamp objects. So you will have to setup additional stand-in objects that can be reflected, such as spheres for standard Lamps, planes for Area Lamps. even emitting monkeys, or whatever else may be needed.

You will have to place these objects at the same location and orientation of the Lamp objects, and on another layer (if you put a sphere at the exact location of a Lamp with shadows turned on, then of course there will be shadows everywhere, since the light would not even escape from within the sphere)

by Victor Malherbe

Instead of entering the details of the parameters to use, I did a screenshot of the values I used to obtain the next render. Thirty-two samples are a good compromise between render time and quality. For final renders, you should use 64 or even 128 samples. Beyond these values, the noise is so unnoticeable that we can barely see the difference.



I think that's all we can say about soft reflections. Now all you have to do is insert your new shader into the nodal tree of the skin shader, and screen it above the diffuse part of the shader. If you want more realism, you might want to use two layers of soft reflections so as to replace the old speculars. It would be logical, since we also called two specular terms in our original nodal tree.

2. Improving the Shader Further

The shader we have been building is a “generic” material, which is intended to fit a lot of situations. And yet, it can be customized depending on what kind of skin you want to render. For instance, you might want to see "fuzzy" skin. Then your material should be soft looking, with a kind of Fresnel effect, simulating the countless strands which appear more obvious when seen at grazing angles. Such an effect can be done by playing with some “Normal” and “Geometry” nodes in the Material Nodes editor.

Or, if you do not want to do it with nodes, you can simply use a “Minnaert” diffuse shader instead of the Oren-Nayar shader I chose to use for the un-scattered diffuse. Note that the “edge enhancement” effect can also be done using raytraced soft reflections with some Fresnel. It's up to you to tweak the material to your liking: what I propose is just a "skeleton", a base concept that you can modify and adapt as desired.

Another idea to explore: when building our nodal tree, all mixing factors were fixed, but you might want to use textures to control these “Fac” values. For instance, the "Screen" factor between the dermis and the epidermis is set to 0.6, whereas it is not constant in the reality: under your feet, the skin appears yellow because the epidermis is quite thick; the lips appear almost red because the epidermis is thin in that precise area, and so on.

Nothing will prevent you from painting a texture which defines the thickness of the epidermis, and then to insert it into your nodal tree so as to obtain the desired effect. Or, you might even just allow the red values of the color texture give you a "fake" epidermal thickness.

3. Some Rendering Tips

In conclusion, some rendering/post-processing tricks may be in order to increase the realism of your skin renders.

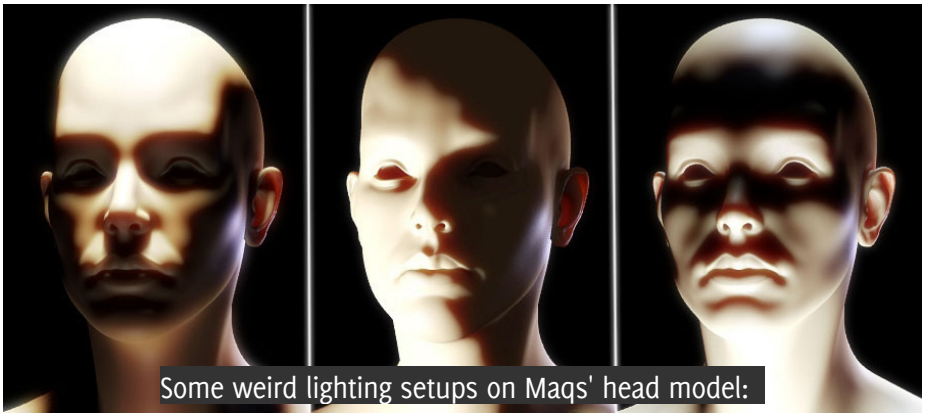
The first trick is quite simple: so as to enhance the contrast/lighting of your render, you can use the overlay node in post-processing, by overlaying the render with itself. Of course, the colors can be tweaked using RGB Curves, some contrast corrections, and so on.

Another piece of advice I would give is to use Ambient Occlusion! Indeed, skin tends to enlighten itself. The closer two areas are, the more saturated (orange, in the case of skin) they will look, because of the numerous bounces of light between those areas. And what is great with AO is that to some extent, it provides “distance values” of the rendered polygons. You simply need to adjust your AO pass so as to be orange, and then to multiply it to your render, just like on the image below. You will notice for instance, that the eyelids and the lips are better “drawn”, so to speak.

Conclusion

In the end, we have achieved a quite realistic shader which will, I hope, help you to get rid of the commonly seen plastic-looking skin. Firstly, I would like to thank the Blender community, which gave me a lot of helpful advice . and above all, thank you for reading. All I need to wish you now is "Happy Blending!"

Below are some test renders, showing what can be done with this shader. No post-processing was used, except for the glow.



Some weird lighting setups on Maqs' head model:

by Victor Malherbe

by Victor Malherbe

The ever-useful Stanford Buddha with a fleshy look, and his friend the Stanford dragon:

By François Grassard aka CoyHot

Introduction

Like any other atmospheric effects, creating realistic smoke is not an easy task. Every kind of smoke requires different techniques. If you need to simulate cigarette smoke, check out the wonderful tutorial of Meltingman in this Blenderart's issue. It's a really easy way to do it with convincing results. But when you need to simulate a more dense smoke, you have to choose another technique.

Of course, the best tool you could use for that is based on fluid simulation for creating motion and voxels for rendering. For instance, Lightwave provides a powerful tool named «Hypervoxel» to achieve this kind of effect. Because Blender doesn't have those features and because voxels rendering can be a time consuming process, we have to choose another method.

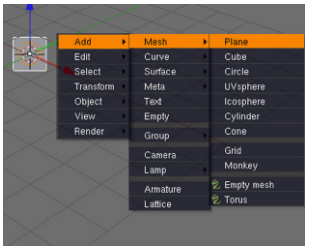
For several years, CG artists working in the special effects' industry have been using particle systems. Controlling the particles flow is not the hardest part, the most difficult part is rendering it. How many particles do you need to create realistic sand movement like the Spiderman 3 scene «Birth of Sandman»? Another example : you have to control billions of particles when you want to simulate an avalanche, how do you render each snowflake and give it the right shading?

For all these kind of effects, such as clouds, smoke, tornadoes, waterfall, sand tempest, burning meteorite, plane crash, explosion, and more ... you can used a wonderful technique called «Sprites» ... or «Billboards» in Blender. Let's see how to use them.

A) Creating the emitter and textures :

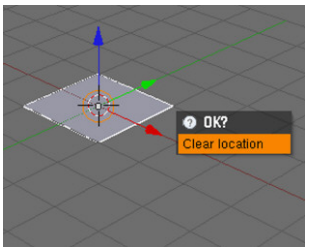
Step 01:

Create a new Add ›› Mesh ›› Plane, then leave the edit mode using the "Tab" key if needed.



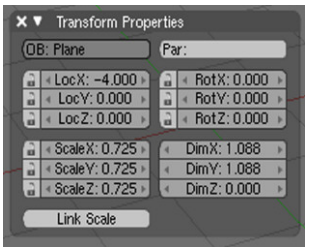
Step 02:

Press ALT+R and ALT+G keys to place the plane at the center of the world.



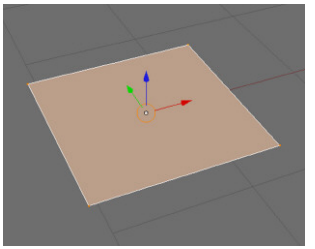
Step 03:

Press the N key to show "Transform Properties" in a floating window. Set Scale X, Y and Z to 0.725 and Loc X to -4.



Step 04:

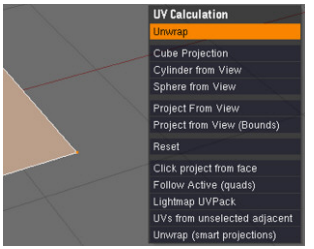
Press the Tab key to switch to edit mode and press the A key to select all vertices.



by François Grassard

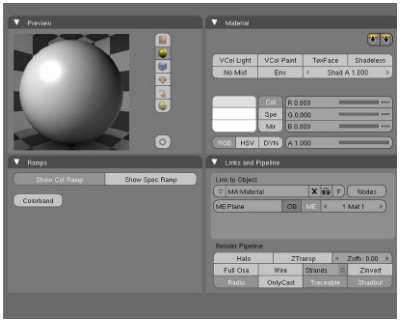
Step 05:

Press the U key and choose "Unwrap" to generate UV coordinates. If the emitter does not have UV coordinates, Billboards won't be displayed correctly. Press "Tab" again to exit the edit mode.



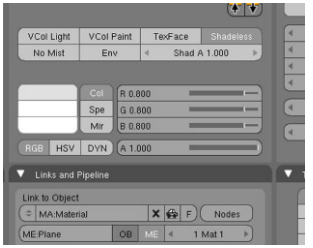
Step 06:

In a "Buttons Window", press the F5 key to switch to the "Shading" panel, select the plane and click on "Add new" to assign a new shader to your object.



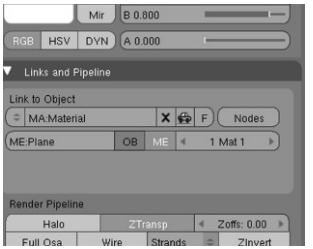
Step 07:

In the shading editor, check "Shadeless" to make your smoke visible, even if there's no light in your scene.



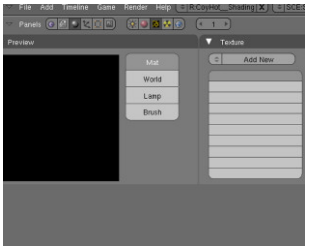
Step 08:

Check the "ZTransp" button to activate transparency without using raytracing and set the A ("Alpha") value to 0, using the texture's transparency instead of "global" alpha.



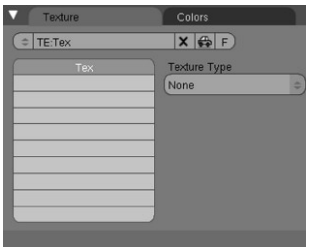
Step 09:

Press the F6 Key to show the "Textures" panel.



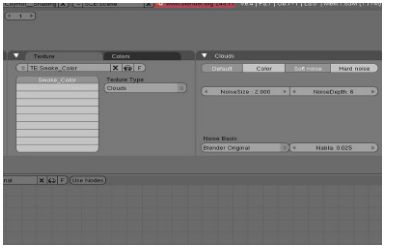
Step 10:

Click on one of the blank buttons and press "Add New".



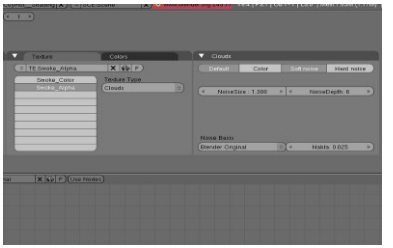
Step 11:

Set the texture's type to "Clouds" and set "NoiseSize" to 2, "NoiseDepth" to 6 and rename your texture to "Smoke_Color".



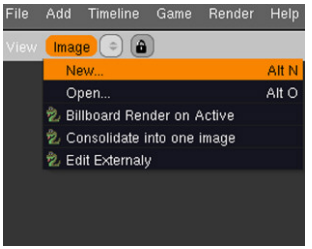
Step 12:

Once again, click a new blank field, press "Add New", choose "Clouds" as the type and "NoiseSize" to 1.3 and "NoiseDepth" to 6. Rename this texture "Smoke_Alpha".



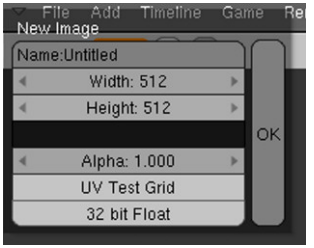
Step 13:

Show the UV/Image editor to paint a new texture and create an Image » New.



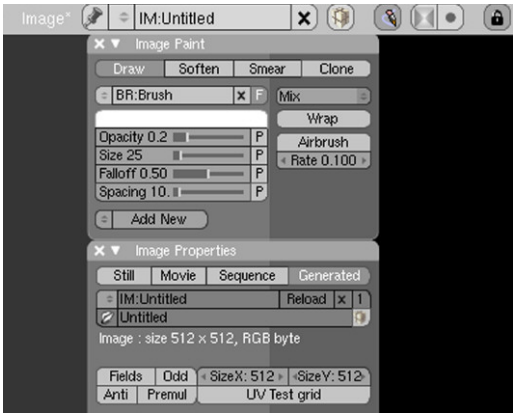
Step 14:

Set "Width" and "Height" to 512, defining the size of the new texture and press OK.



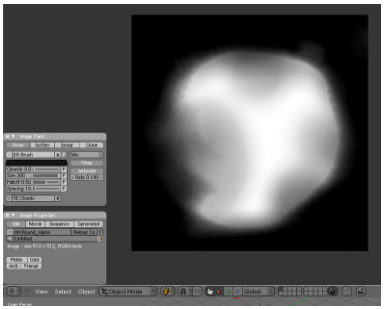
Step 15:

Press the "Enable Painting Texture" button of the header of the UV/Image Editor to activate the "paint" mode and press N and C keys to show the "Image Paint" and "Image Properties" palettes.



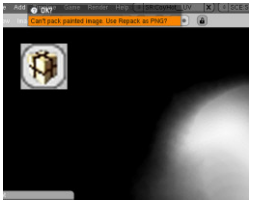
Step 16:

Use the paint tool to create a random white shape, but don't touch the edges of the image. They have to remain totally black. While painting you can press the "Airbrush" button to paint continuously.



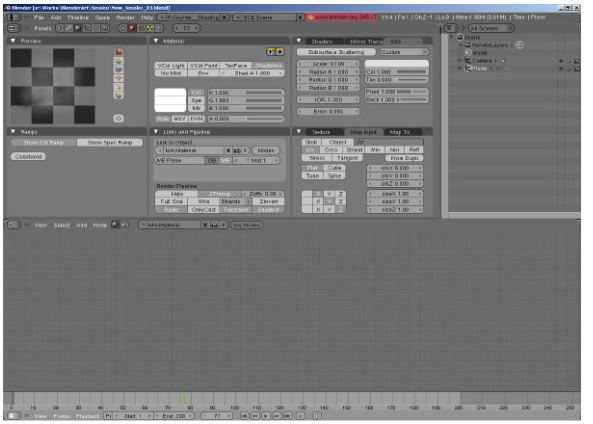
Step 17:

Once your texture is finished, name it "Smoke_Mask" and check the "Pack into blend" button. Blender will ask if you want to repack this texture as a PNG file ... just click on it.



Step 18:

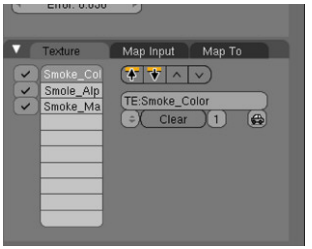
Get back to the shading editor using the F5 Key.



by François Grassard

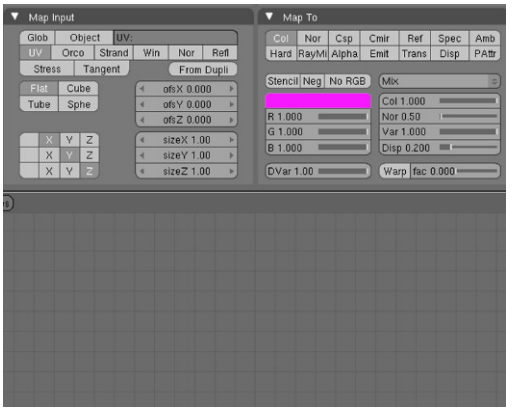
Step 19:

Click on the "Texture" tab, click "Smoke_Color" and switch to the "Map Input" tab.



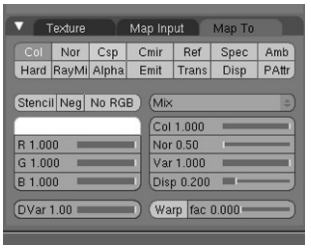
Step 20:

Check the "UV" button, click on the "Map to" tab and check "Color". Uncheck any other button if necessary.



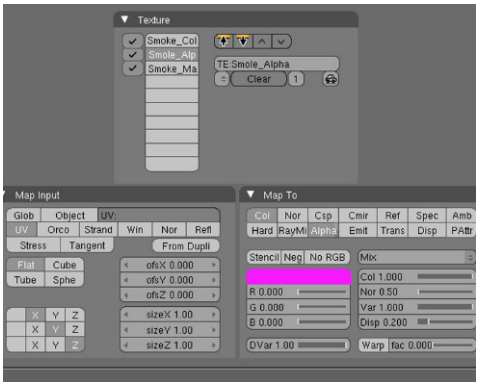
Step 21:

Click on the purple color and set it to white.



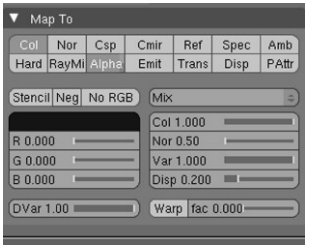
Step 22:

Get back to the "Texture" Tab and select "Smoke_Alpha". In "Map Input", check "UV" then in the "Map to" check "Color" and "Alpha" (be careful, don't check the Alpha button twice otherwise the transparency will be inverted).



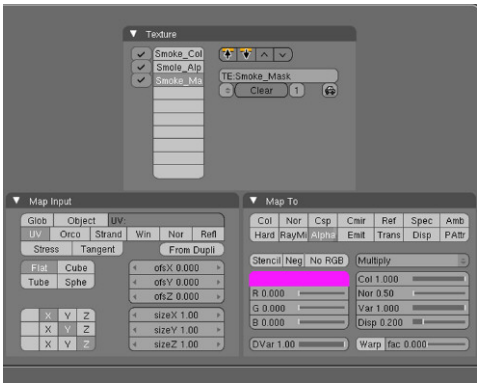
Step 23:

Click on the purple color and set it to black.



Step 24:

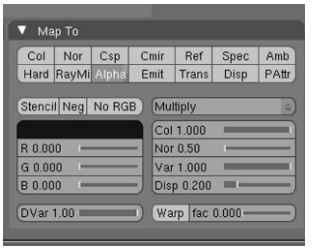
Get back to the "Texture" tab and select "Smoke_Mask". In "Map Input", check "UV" and in the "Map to" tab check only "Alpha".



by François Grassard

Step 25:

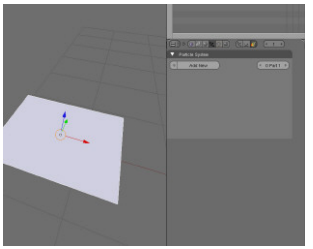
Set the purple color to Black and change the "Mix" mode to Multiply.



B) Animate your particles flow :

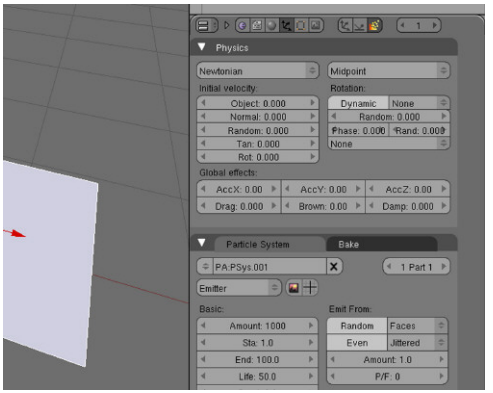
Step 01:

Select the emitter object (here, the plane) and press the F7 Key several times until the Particle panel appears.



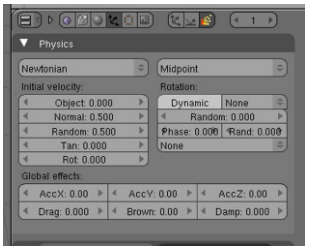
Step 02:

Click the "Add New" button to create a new flow, named «Psys».



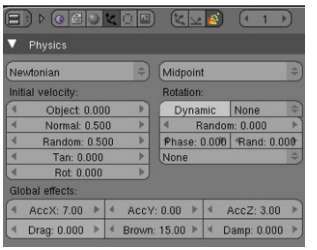
Step 03:

For "Initial Velocity", set "Normal" and "Random" speed to 0.5



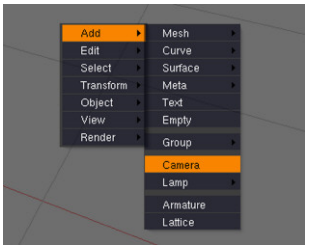
Step 04:

In "Global effects", set the acceleration parameter "AccX" to 7, "AccZ" to 3 and "Brown" to 15 (really useful to have a non-linear motion).



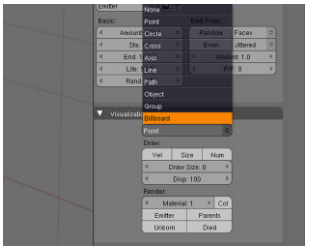
Step 05:

Now you have to create a new camera, because we are going to use the famous particle's type named "billboard".



Step 06:

In the "Visualization" panel, set the particle's type to Billboard. Billboard, aka "Sprite" is a simple square always facing the camera (that's why you have to create a camera to refer to).

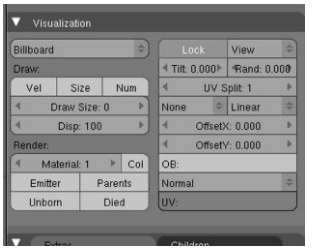


by François Grassard

by François Grassard

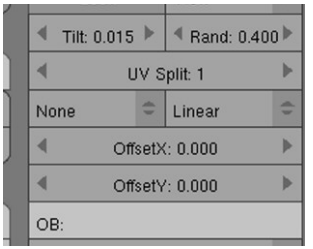
Step 07:

Check the "Lock" button to keep all Billboards parallel to avoid intersection between them.



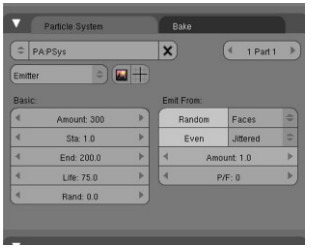
Step 08:

Set the "Tilt" (rotation value, which seems to be expressed in "Radian" unit) to 0.015 and set the "Rand" value to 0.4 to give each billboard a different angle at birth.



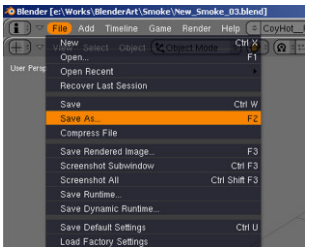
Step 09:

In the "Particle System" tab, set the "Amount" value to 300, "End" to 200 and "Life" to 75 ... or a little bit more, until your billboards go off-screen when you look through the camera.

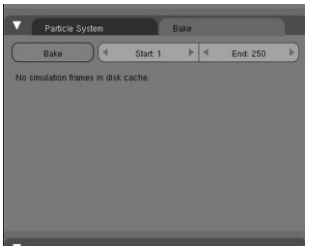


Step 10:

Save your scene somewhere on your hard drive. I'm not kidding, it's really important !



Step 11:



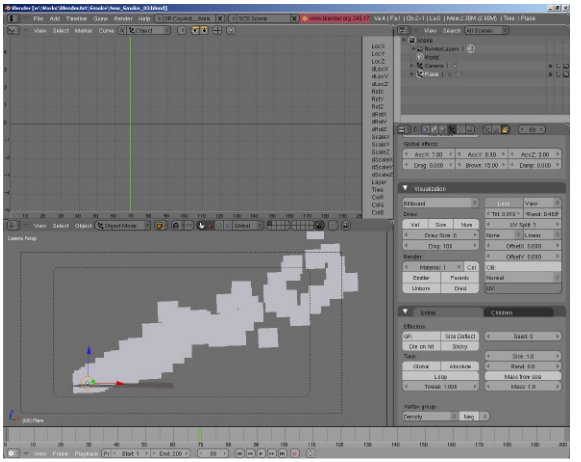
Click on the "Bake" tab, just next to the present one and click on "Bake". A new directory is created next to the .blend file you saved on your hard drive, (do you understand now why it was so important?), filled with a bunch of files (one per frame) where all particles data is stored.

FYI : Once your data is baked (or "cached" in other words), you can still edit all parameters that won't affect the particles' motion (like size, color, transparency, children, ...)

C) Animate the billboards twist and size :

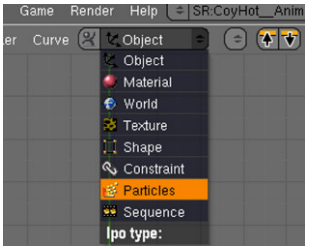
Step 01:

Now that your particles' flow is baked, select the emitter object (here, the plane) and switch to the IPO Editor.



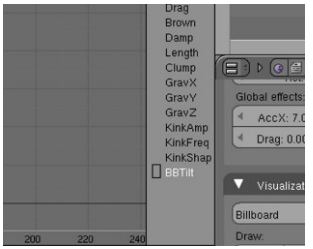
Step 02:

Click on the "Show IPO type" combo list and choose "Particles". Now you can see different particles' parameters on the right of the editor.



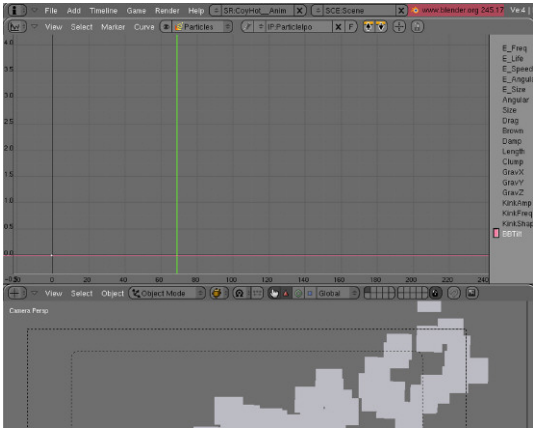
Step 03:

Click on "BBTilt" (meaning Billboard Tilt). This parameter is used to rotate the billboard according the camera view and particles' age.



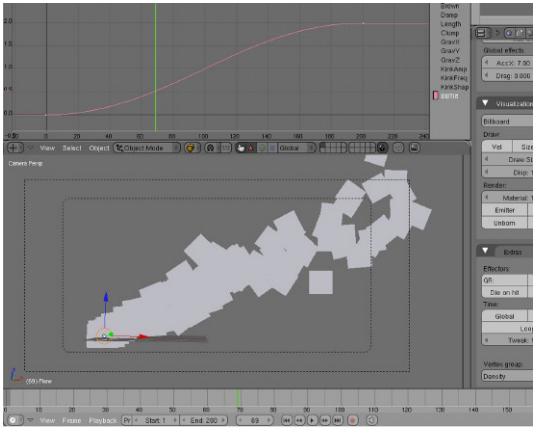
Step 04:

Press CTRL key, and click with the left mouse button around the coordinates 0.0/0.0, adding a new point (use the right mouse button if "Select with Left mouse button" is set in your Blender's preferences).



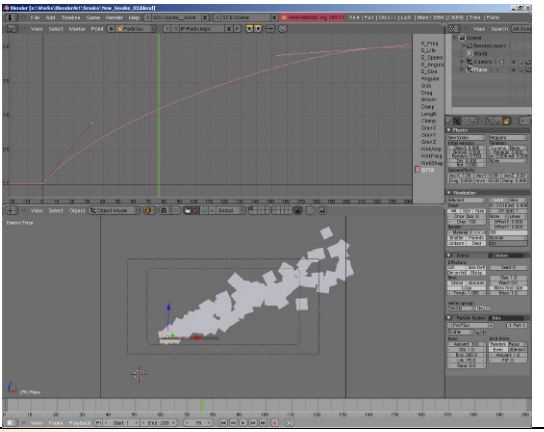
Step 05:

Once again, create a new point around the frame 200 with a value of 2.0.



Step 06:

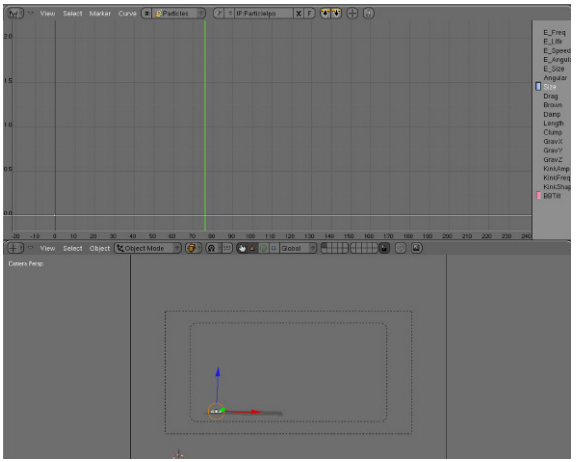
Tweak the tangents' knots to mimic a logarithmic curve.



by François Grassard

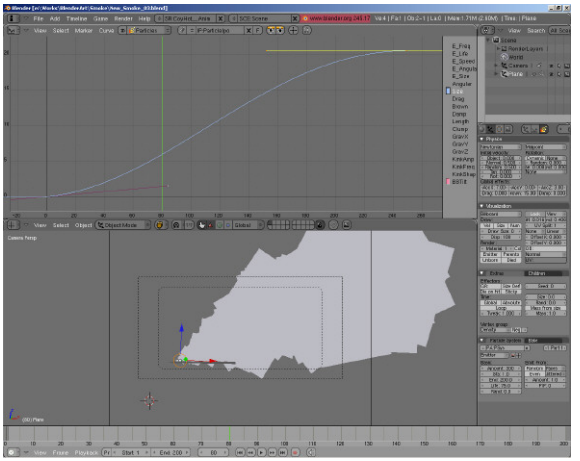
Step 07:

Click the "Size" parameter, on the right of the IPO Editor, and create a new point at 0.0/0.0 as the same way you've done for the "BBTilt" parameter.



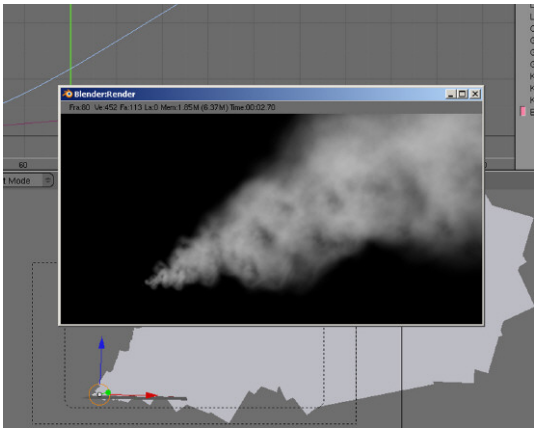
Step 08:

Add a new point around frame 200 with a value of 20. Leave the default interpolation unchanged. Now if you press "Play" you can see your billboard growing and twisting along the animation.



Step 09:

Press F12 to start render and see what happened. Yes, now you have heavy and dusty smoke !!!!!

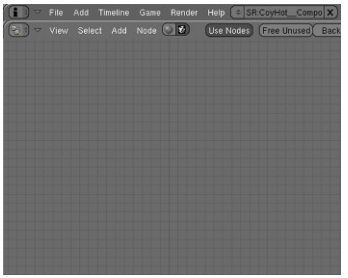


by François Grassard

D) Enhance your render with compositing :

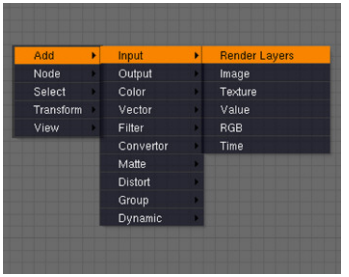
Step 01:

OK, now your smoke looks realistic but looks too flat and needs more details and contrast. So, just open the "Node editor" and press the "Composite Nodes" and "Use nodes" buttons.



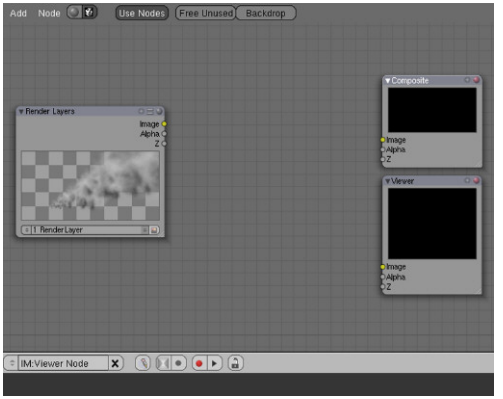
Step 02:

If you have no "Render Layers" in your composite graph, add a new one via Spacebar ›› Add ›› Input ›› Render Layers.



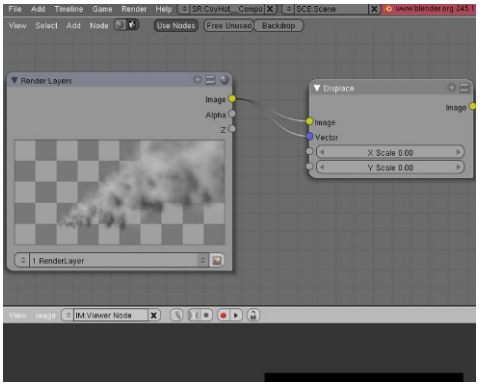
Step 03:

Create a new Spacebar ›› Add ›› Output ›› "Composite" node and a new Spacebar ›› Add ›› Output ›› "Viewer"



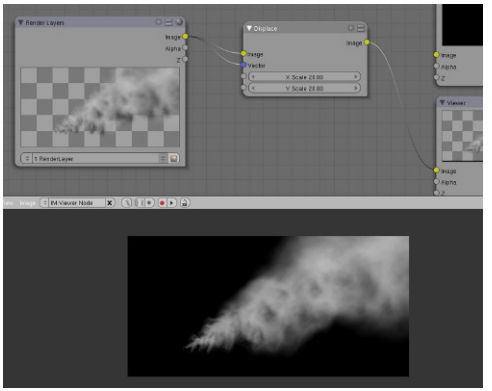
Step 04:

Create a new Spacebar ›› Add ›› Distort ›› "Displace", plug the output parameter "Image" of the "Render Layers" into "Image" and "Vector" of the "Displace" Node.



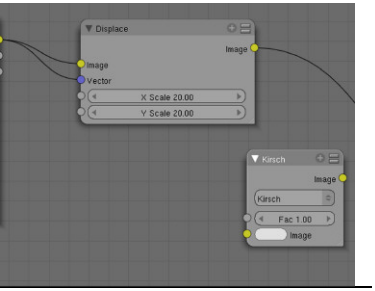
Step 05:

Set the "X Scale" and "Y Scale" of the "Displace" to 20. The Displace is used to give to the smoke a more "fluid" motion.



Step 06:

Create a new Spacebar ›› Add ›› Filter ›› Filter and set it to "Kirsch" mode.

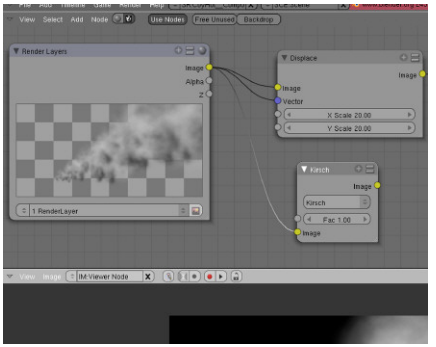


by François Grassard

by François Grassard

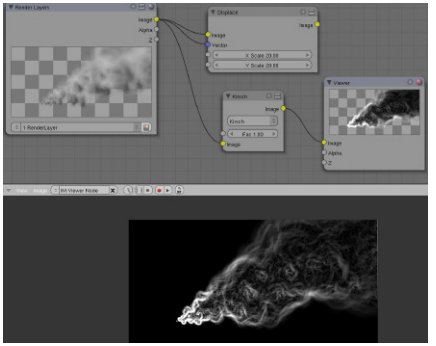
Step 07:

Plug the output "Image" of the "Render Layers" to the "Image" input of the "Kirsch" node.



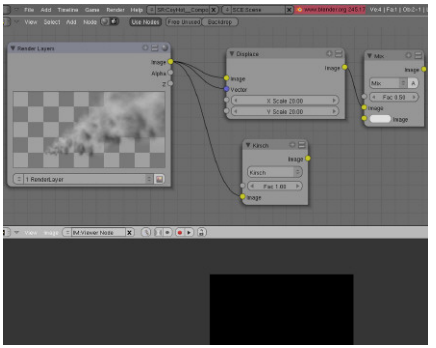
Step 08:

Set the Fac of the "Kirsch" to 1.0 if needed. Kirsch is used to generate additional details that will be mixed to the original render.



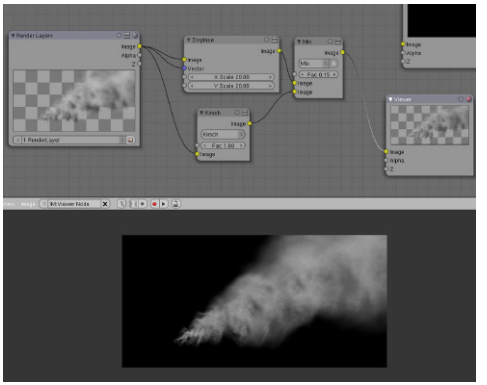
Step 09:

Create a new Spacebar ›› Add ›› Color ›› Mix, plug the "Image" output of the Displace to the first "Image" input.



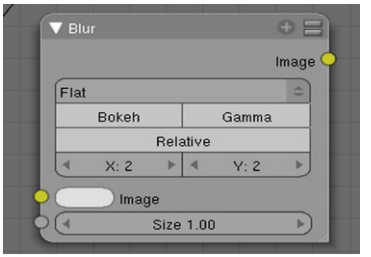
Step 10:

Plug the "Image" output of Kirsch into the second input of the Mix, set the "Fac" value to 0.15 and click on the "A "("Alpha") button.



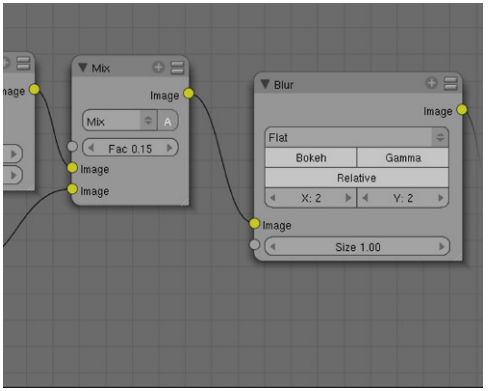
Step 11:

Create a new Spacebar ›› Add ›› Filter ›› Blur and set the X and Y parameters to 2. This Blur is used to give to the smoke a softer look.



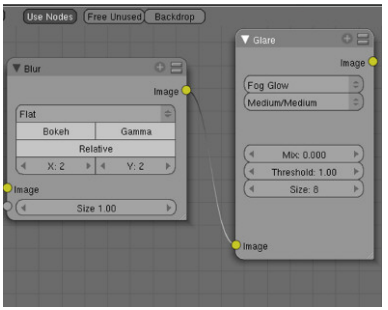
Step 12:

Plug the "Image" output of the "Mix" into the "Image" input of the "Blur" node..



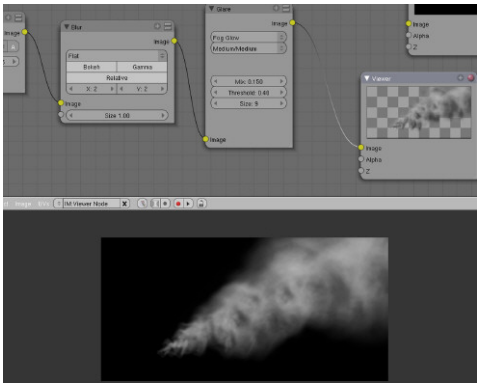
Step 13:

Add a Spacebar » Add » Filter » "Glare" node. Set it to "Fog Glow" mode and plug the output of the blur node into this single Input.



Step 14:

Set the "Mix" value to 0.15, "Threshold" to 0.40 and "Size" to 9. Now your smoke looks more contrasted, visually increasing the density.



Step 15:

To finish, plug the Glare's output into "Composite" and "Viewer".

Step 16:

In the "Buttons Window", press F10 key to switch to render settings and check "Do Composite".

Step 17 :

Set the "output" path where you want to save your rendered frames and press "Anim" to render your animation.

Conclusion

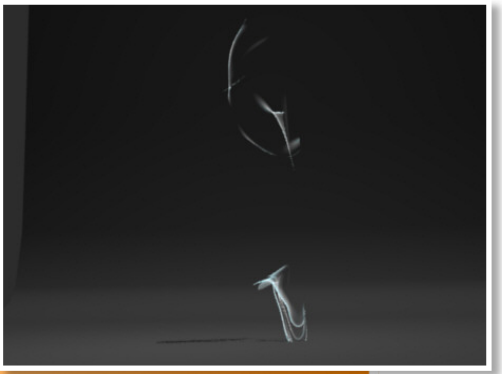
With Billboards, you can achieve a lot of wonderful and realistic effects. A standalone piece of software called «Particle Illusion» (http://www.wondertouch.com) is a sprites' based special effects tool, working exactly the same way as Blender's Billboards ... but in realtime (I'm waiting for all the developments created by the Apricot project, especially GLSL support in the 3D view that will make possible the visualization of the billboards in realtime, with shading and alpha).

Take a look at this incredible tool ... it will certainly give you some ideas of what is possible with this technique. The next step is to provide Blender the same kind of libraries available with «Particle Illusion» to easily create those effects, like a scene containing several Psys (smoke, waterfall or fire), accessible by any emitter in the scene and imported via File » «Append or Link».

Make your own libraries and share them with everyone :

All the particles effects could be collected in a single website, like www.blender-materials.org for instance and available for everyone. Together, we can make it ! So, lets play with billboard and particles and enjoy !!!

by François Grassard



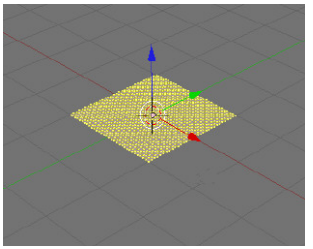
By Georges Mignot aka Meltingman

Introduction

This tutorial will teach you how to create a twirl of smoke in blender.

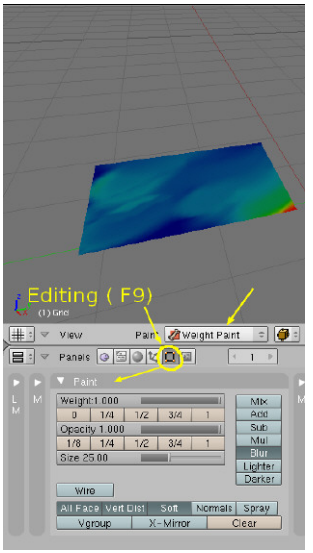
PART I: The Actors of the Sequence

Step1: First, launch Blender, erase the default Cube and then add a Grid.



Space Bar › Add › Mesh › Grid (32 x 32). Go back to Object Mode : Tab.

Next, with the grid selected, switch to Weight Paint mode: Ctrl + Tab. Then paint your grid in order to get only one reddish orange point, and the rest from yellow to blue.

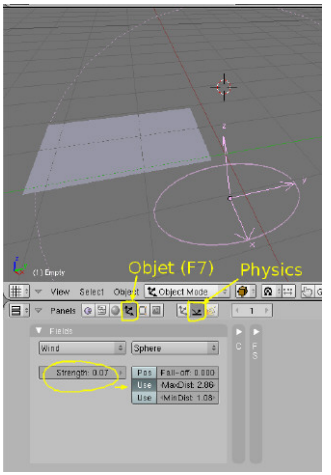


Use the Paint tab, in the Edit panel (F9).

- Mix, Add and Multi › used to mix, add or multiply the painted weight.
- Sub › allows subtracting some weight.
- Blur › very useful for smoothing the weights.

Step2: Add the deflector object.

(Deflectors interrupt the movement of active particles, Soft Bodies, etc.)



Place an Empty (Add › Empty) just below the corner of the Grid where the weight is at its maximum.

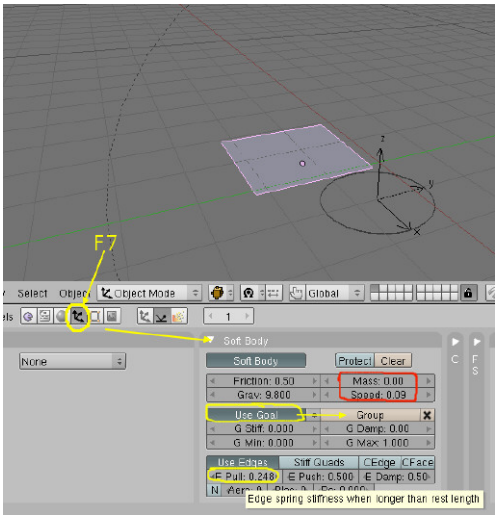
Enable the Deflection, Physics Button (F7), then set it up like this :

- Wind with a low Strength : between 0.010 and 0.020
- Enable Use Max Dist, then type the maximum distance you need. This will limit the wind's influence to the area within this distance.

by Georges Mignot

Step3: Give some softness to our Grid.

Thanks to specific adjustments to Soft Bodies, we will achieve the illusion of a soft and aerial material.



Select the Grid, open the Object panel (F7), and the Physics sub-panel.

Next, access the Soft Body tab and set it up like this :

- Mass to 0 (or, extremely low).
- Very low Speed (tip: you can increase it for the preview then decrease it for the final render).
- Enable Use Goal › Group (to take your painted vertex weights into account).
- Enable Use Edge and decrease it to the halfway mark (the edges can now stretch according to the needs of our simulation).

Now, for the testing step.

In order to launch a preview animation in our 3D window, set the view to the active camera (0 Numpad key), then press Ctrl + A while in the 3D window.

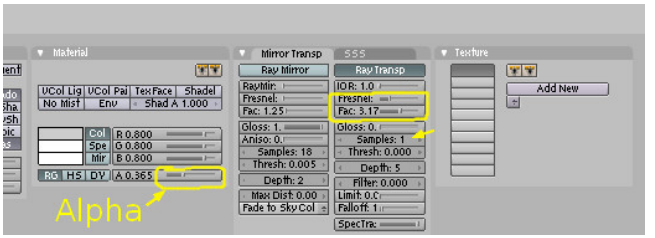
Adjust the Soft Body or deflection settings accordingly.

PART II : Ethereal Material

Our settings should simulate a fairly realistic smoke, with fast render times.

- Give a material to our Grid.
- Shading panel (F5), then Material tab,
- Add New, then test the result (F12) while setting up the options.

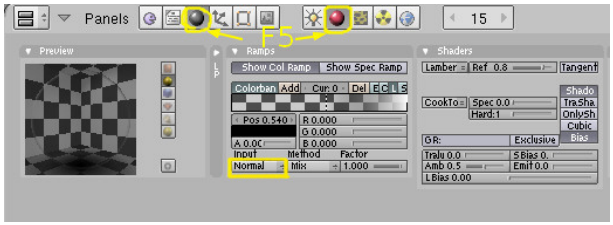
In the Mirror Transp tab, activate the Ray Transparency with:



- IOR at 1.0 (no deformation due to refraction of light)
- a bit of Fresnel, moving the Fact slider to lower it's influence
- decrease the Gloss to 0 in order to skip some useless computation, and set the Samples to 1.0
- Depth value will be between 5 and 10 (depth of transparency).

Georges Mignot
France

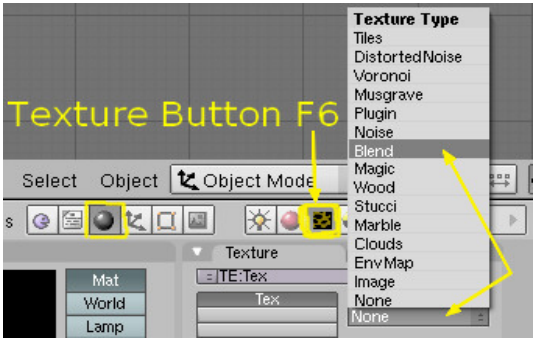
Webpage



- In the Material tab, decrease the Alpha to between 0.0 and 0.25
- Open the Ramps tab (next to the Material tab) and enable the Show Col Ramp button
- Set a blend from white to black, with Normal as input for ramp, which will give you a silky effect.

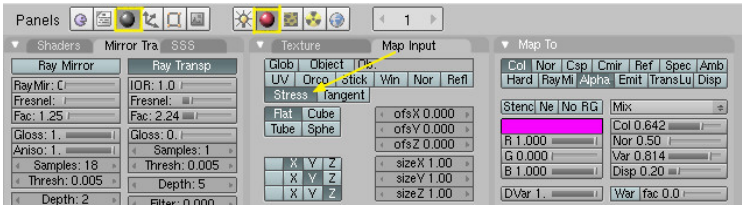
A "new" function can be used here.

- The Map input › Stress button will vary the blending of a texture according to the deformation of a mesh.
- So, add a Blend Texture in the Texture panel (F6) › Add New › choose Blend in place of None.



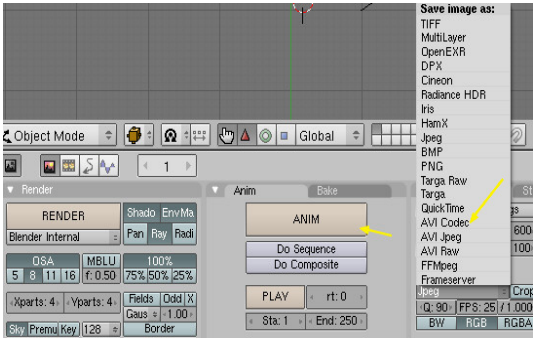
Go back to the Material panel.

Add this texture with Map Input › Stress with Map To › Col and Alpha buttons checked.



Now, you are ready for the final render. You can start the rendering of the sequence:

In the Scene panel (F10) choose your file format (AVI or QuickTime for video, for example), and finally, press the Anim button to start the rendering.



by Georges Mignot

By Slavoljub Pantelic

Introduction

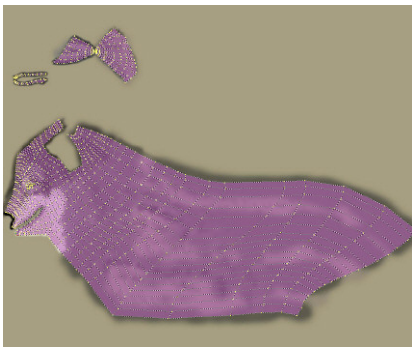
In this tutorial you will see one simple method of making fur for wolves or similar animals, suitable for static images.

Step 1

First, try to find some reference images with front, side and top views of a wolf's head with a pose similar to a blueprint. One method is to make the head using box modeling, a Mirror modifier and a Subsurf modifier set to 1. Once you are satisfied with the basic model, you can apply the Subsurf modifier, after which you can make fine corrections to your model.

Step 2

After marking your seams, UV-unwrap the mesh. Export the UV layout into an image editor (e.g. GIMP, Photoshop, etc.) and then paint over the layout on separate layers. This will be the base skin of the animal. Because this skin is under the fur you don't need to worry about being precise.



Step 3

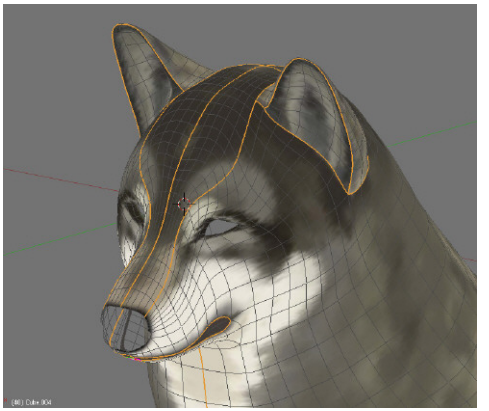
Find some close-up images of fur and adjust it in your image editor with the brushes, filters and other tools. These images will define the tone of the fur's color. Again, precision is not essential; the point of this step is the basic color.

Step 4

Separate the nose, make the eyes and duplicate the mesh. Separate the nose, make the eyes and duplicate the mesh.

Step 5

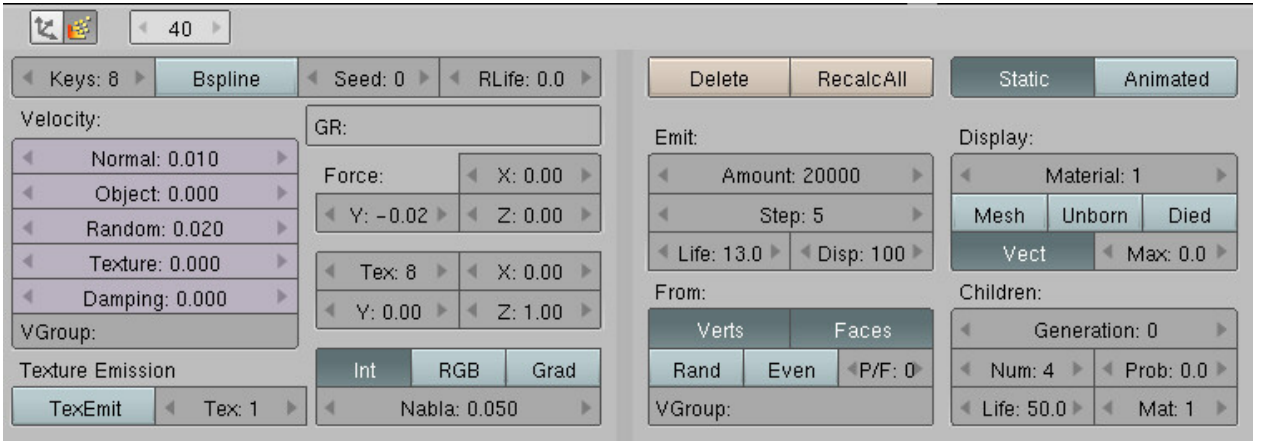
Divide the duplicated mesh into separate parts where the fur markedly changes and then slightly extrude edges on each fur surface to cover the neighboring surface. This will help the the various furred areas blend to look more natural.



Step 6

Add different fur textures to each part and modify the particle settings according to the natural fur colors. For the best result, add some spherical and vortex deflection for a more natural look to the shape of the fur. With the node editor you can apply some DOF with a bokeh effect for the background, and then put your "pet" in his natural environment!

Below are the particle setting for the hairs used for the wolf model. And On right it the progression for separating and modifying/readying the mesh for particles.



by Slavoljub Pantelic

By François Grassard aka CoyHot

Introduction

Simulating an ocean is one of the case studies that all cg artists will try to work on one day. Several techniques are available, depending of what kind of software you use. Using fluids is obviously one of the best ways to create this kind of effect, but it's also the most time consuming. For the boats' fighting scene of «Pirate of the Caribbean III», thousands of computers computed over several weeks to create the fluid simulation cache for the ocean.

Another method is to simulate the motion of an ocean using «regular» techniques, such as modifiers and keyframes. That's what we describe here, focusing on the waves' motion and not the shading (two .blend files of the three provided contain really basic shading).

Last thing before we start: all values are only given as a rough guide and may have to be adjusted, depending of the size of your ocean and, in general, the scale of your whole scene.

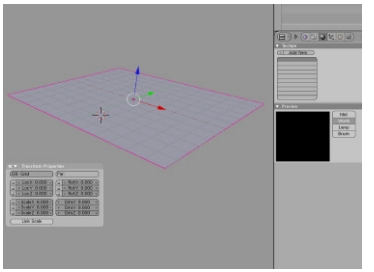
A) Prepare the three basic textures :

Step 01:

Once our dear Blender is started, create a new grid (Add » Mesh » Grid). Set X Res and Y Res to 64 and press OK. If Blender automatically enters edit mode, press Tab to switch back to object mode.

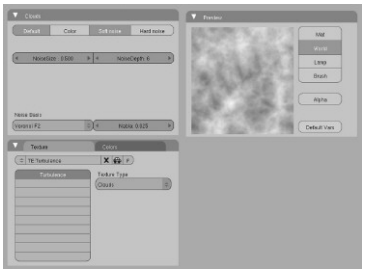
Step 02:

Press ALT+R then ALT+G to reset the location and rotation values, moving the grid to the center of the world. Press N key to display the "Transform Properties," and set ScaleX, Y, and Z to 6. Press F6 to switch the «Buttons Window» to the «Texture» panel and press the button named «World».



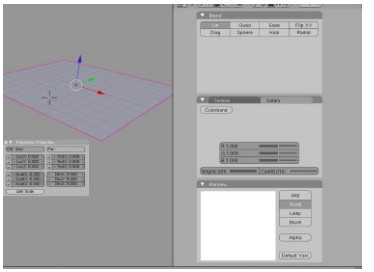
Step 03:

Click "Add New" to create a new texture. In «Texture Type», switch from «None» to «Clouds» and set its «NoiseSize» to 0.5, its «NoiseDepth» to 6 and its "Noise Basis" to the "Voronoi F2" type. To easily find this texture afterward, change its name to «Turbulence».



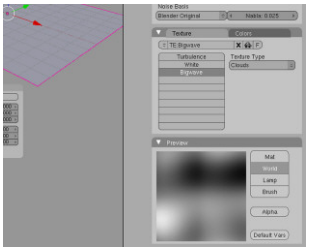
Step 04:

Click the blank button just below «Turbulence», press «Add New» again and choose the «Blend» type. Click the «Color» tab, just next the «Texture» tab. Set the «Bright» parameter to 2.0 and Contrast to 0.01. Click the «Texture» tab once again to rename your texture : «White».



by François Grassard

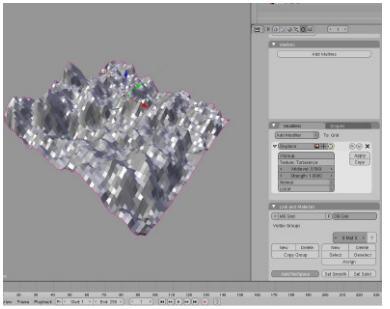
Step 05:



Click a new blank button, press «Add new» once again and choose «Clouds», set his «NoiseSize» to 1.0 and his «NoiseDepth» to 0. Name this texture «Bigwave». Now you have your three basic textures to create the ocean. Press F9 key to switch to the «Editing» panel and select the grid.

B) Ready, steady, displace !!

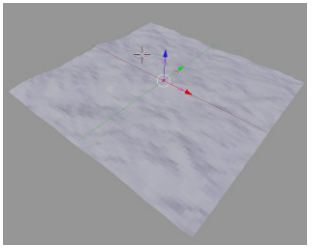
Step 01:



Now, it's time to use the three basic textures you previously prepared. For that, you will use one of most powerful modifiers provided by Blender, the «Displace». Select your grid, press the «Add modifier» button and choose «Displace». This modifier is designed to deform a mesh according the luminosity of any texture. In the displace's parameters, set the «Texture» field to «Turbulence», referring to one of the textures we previously made.

Step 02:

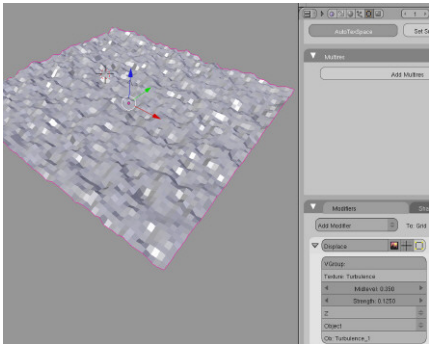
In the displace's parameters, set "Midlevel" to 0.35 and "Strength" to 0.125, decreasing the effect's influence . To animate this elevation, we need to use an object to tell the displace modifier the offset between the texture and the grid. To do that, create a new Add ›› Empty then press ALT+G and ALT+R to place it at the center of the world.



Step 03:

Select the Empty and press F9 Key to Switch to the «Editing» panel. Rename this Empty «Turbulence_1». Select the grid and inspect the Displace's parameters in the «Editing» panel. Click on «Normal» and switch to «Z», then click on «Local» and switch to «Object».

Step 04:

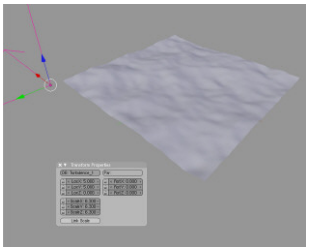


«Object» mode is used to add an offset in the displacement according to the position of another object. In the «Ob:» field, type «Turbulence_1», referring to the Empty you previously created. If you want to test the way the Displacement works, try to move this Empty.

by François Grassard

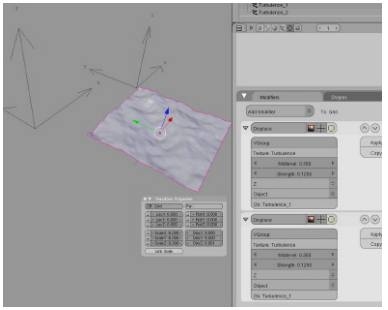
Step 05:

Select the Empty and press N Key to show «Transform Properties» in a floating window. Set Scale X, Y and Z to 6.3. Set LocX and LocY to 5 and set the LocZ value to 0. Select the grid and then click the «Set Smooth» button in the «Editing » panel.



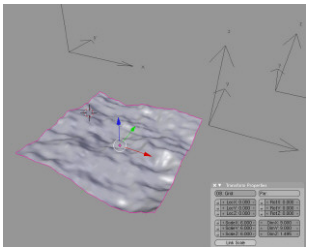
Step 06:

Create another Empty, name it «Turbulence_2» and place it at the center of the world via shortcuts Alt+R and Alt+G, as before. Set their LocX, Y and Z to -8.5 / 6 / 3 and the three Scale values to 6.5. Select your grid, then go back to the «Editing» panel and press the «Copy» button of the «Displace» to duplicate this modifier.



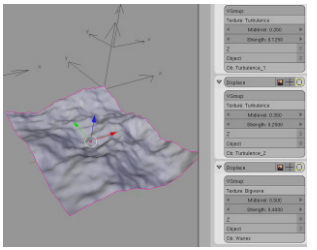
Step 07:

Change the «Strength» value of this duplicated «Displace» to 0.25 and set the «OB:» parameter to «Turbulence_2». The two «Displace» modifiers now can be controlled independently. Create a new Empty and place it at the origin of the world (Alt+R and Alt+G). Name it «Waves», set LocX, Y and Z to 9 / 9 / 2 and the three scale values to 4.5.



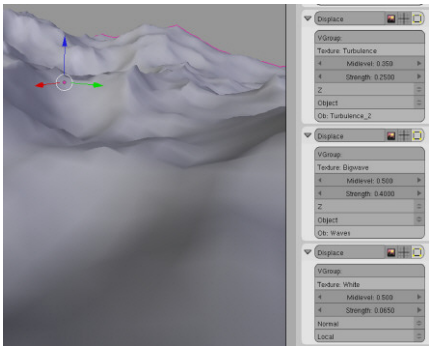
Step 08:

Select the grid and, as before, press the copy button of the lower «Displace» modifier in the stack. Now you have three «Displacements» applied to the grid. Two for adding small turbulences on the ocean's surface and the lower one in the stack to create bigger waves. Fill his «Texture» field with «Bigwave», referring to another texture we previously created. Set the «Midlevel» value to 0.5, «Strength» to 0.4 and «Ob:» to «Waves», referring to the last Empty created.



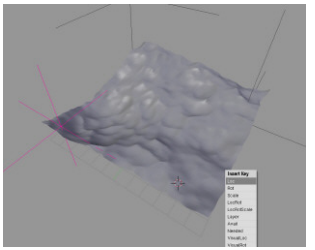
Step 09:

Once again, click the «Copy» button of the lower «Displace» modifier in the stack. For the parameters of the fourth «Displace», click on «Z» and switch to «Normal». Next, click on «Object» and switch to «Local». Set the «Strength» to 0.065 and fill the texture field with «White», on of the texture you previously created. This fourth «Displace» modifier is just here to push all vertices according there own normals and for generating the crests.



by François Grassard

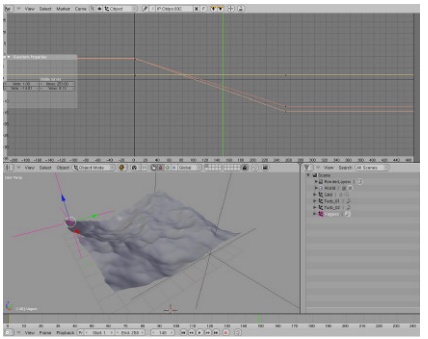
C) Animate waves and add floating object :



Step 01:

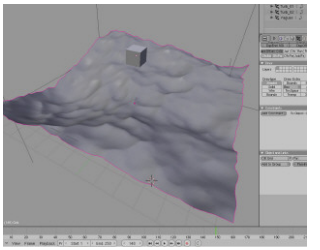
Go to the first frame of the animation, select the Empty «Turbulence_1» and press the I Key to create a new «Loc» keyframe. Go to frame 200, move the Empty to -6 / -4 / 0 using the floating «Transform Properties» window. Press the I Key again and choose «Loc», creating a second keyframe.

Step 02:



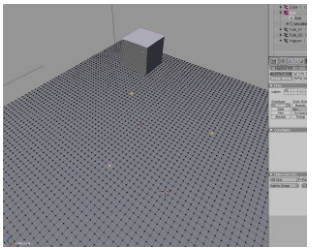
Animate the Empty «Turbulence_2» in the opposite direction using two keyframes between frame 0->200 too. The motions of the two Empties now create an X, adding realistic turbulence on the surface of the sea.
Now, animate the Empty named «Wave», slower than the two others, to generate bigger waves. According to the velocity of the different Empties, you can tweak the waves' motion. Be careful of the default «Bezier» interpolation type of any kind of animated object. A linear interpolation could be more suitable and can be set in the IPO curve editor by selecting all control points and pressing the T key.

Step 03:



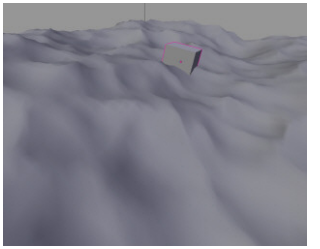
Now that we have our moving ocean, it's time to see how to add a floating object on it. For that, Blender provides a really easy to use function. Add a new cube to your scene, press Alt+R to reset the rotation value to 0, keep the cube selected then press the SHIFT Key and click on the Grid to add it to the actual selection.

Step 04:



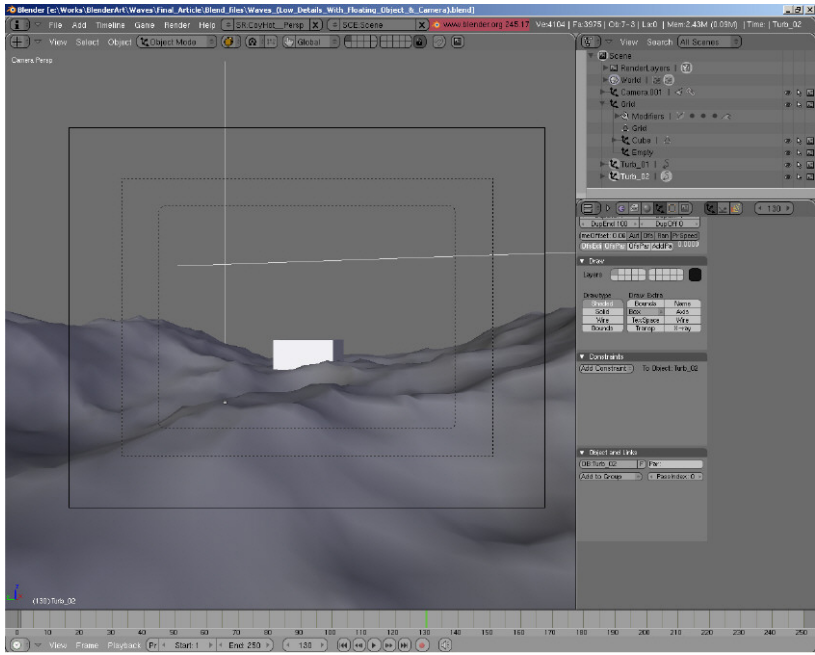
Press the Tab Key to switch the grid to Edit mode. Select three vertices on the grid, forming an equilateral triangle. The bigger the triangle is, the smaller the turbulences that will move the cube. Make a choice according the kind of motion you want have for the cube.

Step 05:



Keep all elements selected and press CTRL+P Keys, then confirm the «Make Vertex Parent» when Blender asks you. Press the Tab key to turn off the Edit mode. Now, you will certainly have to manually replace the cube on the surface using the G key to reduce the size of the dotted line to 0. Once that is done, press play and enjoy!

Optimizations for bigger scene



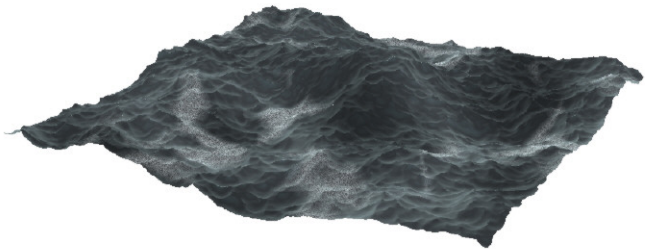
As you can see, this kind of technique is pretty simple to setup and gives convincing results. Displace in Blender is one of my favorite modifiers, because even on a model with thousands of polygons, it's pretty fast (for me, probably the fastest I have ever seen among all the 3D software I have used over 15 years).

Okay, just a last trick before we go. Here, we have only simulated a small portion of an ocean. If you want to create a bigger surface with a high number of subdivisions on the basic grid, you risk having your computer die before you have time to press play button.

But, the subdivision can be dynamically added/removed during your work. Just add a Subsurf modifier in front of the stack in «Simple Subdiv» mode. So, you can increase or decrease the number of subdivisions when you want. Just push the subdiv number higher when you have to tweak the fourth displace modifier (the one that generates the crests), and get back to a more usable number once you have set it up. You can even set two different subdiv levels, one for the 3D view, and the other one for the render engine.

If you can test the latest SVN version of Blender with the «Adaptive Subdivision» patch included (down-loadable on graphicall.org), you can even dynamically reduce the number of polygons according to the distance of the camera.

Far parts of the ocean won't be subdivided enough to reproduce the small turbulence on the surface (but you don't care, because they are to far to be seen), but the bigger displace «Big_Waves» will still work on it ... and the number of polygons for your computer to manage will be dramatically reduced!!! So, you can generate an infinite ocean without a big amount of polygons. Have fun, boys and girls!



by François Grassard

By François Grassard aka CoyHot

Introduction

Creating realistic fire has been one of the biggest challenges of my CG artist life. For many years, I have tried several techniques to simulate this kind of effect. After using simple animated objects with an appropriate shader, particles, sprites, fluids, and even Meta Balls with a special compositing setup (similar to the one used in Shrek), I finally reached my goal of creating the perfect animated fire.

To begin with, I have to say that the initial technique is not mine. The first person to introduce me to this method was the great Alan McKay, one of the best particle artists ever. Last year I wrote a tutorial for the French edition of "Computer Arts" about Ghost Rider and how to mimic the same kind of fire using 3ds Max and especially Particle Flow. I've watched all of Alan's videos, including the one that shows how to create fire using Pflow.

The motion of flames was realistic enough, but the render still looked too artificial. I took my render, dumped it in After Effects and tried to tweak it with several distortion effects and color correction tools. At last, I reached my goal and created the most realistic fire I have ever made.

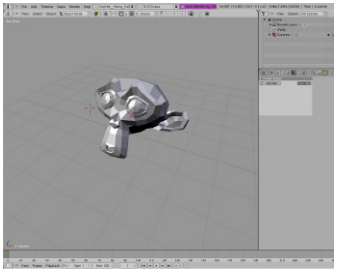
When I finished my tutorial for the magazine, I discovered the first build of Jakha's particles patch on Graphicall.org. I read all of the commit logs and was really impressed by all of the features provided by the patch. Everything was there to create the same kind of effect using only Blender, as a complete compositing tool is included in the software with all of the necessary effects (particularly Vector Blur and Displace nodes).

For fun, I spent several nights to create again - using only Blender - all of the things I've done in 3ds Max and After Effects. This is one of several examples proving that Blender can be used for all kinds of special effects tasks, even on really complex shots. So, ready to burn? Here we go!!!

A) Create the emitter and vertex groups:

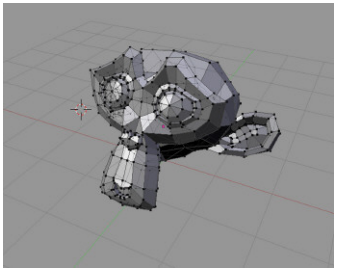
Step 01:

First, we need an object to burn. To start, add a new primitive to your scene, such as Suzanne (the Monkey) for instance.



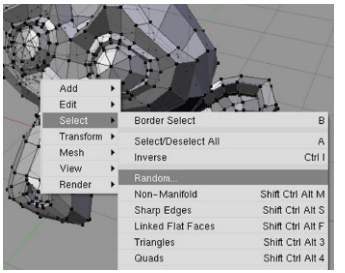
Step 02:

Switch to Edit mode using the Tab key and deselect all vertices using the [A] key.



Step 03:

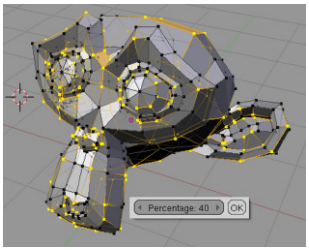
Press the Space Bar on your keyboard and choose Select ›Random.



by François Grassard

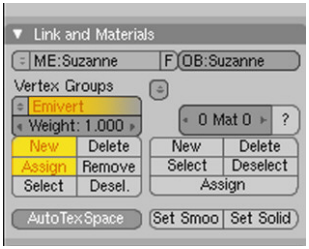
Step 04:

Set the random value to 40% and press "OK". Several vertices are now randomly selected.



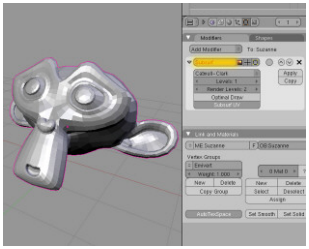
Step 05:

Once your vertices are selected, you have to create a new Vertex Group. For that, press [F9] to show the Editing parameters, and in the Link and Materials panel press the "New" button to create a new Vertex Group. Name it "Emivert" and then click on "Assign", just below the "New" button.



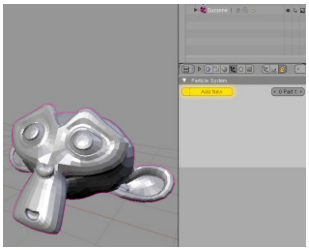
Step 06:

Press the Tab key to exit Edit mode, select Suzanne, and press [Shift+0] to add a Subsurf modifier to it.



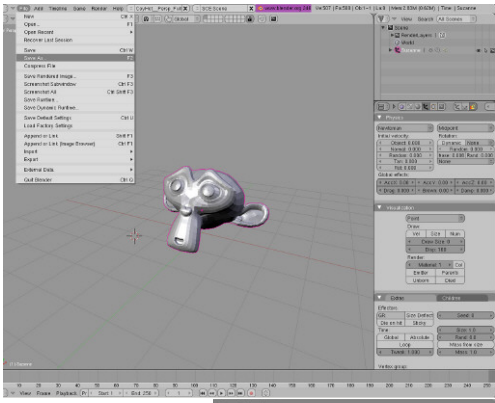
Step 07:

Keep the Suzanne mesh selected and press the [F7] key several times until the particle panel appears in the Buttons window. Press "Add New" to add a new particle system (a.k.a. "Psys") to your object.



Step 08:

Now you have to save your scene somewhere on your hard drive, because Blender will create a new directory next to your ".blend" file where the particle data will be cached.



Step 09:

Keep Suzanne selected and go to the "Physics" panel. Set "Random" to 0.5, "AccZ" to 8.0 and "Brown" to 5.0. The particles now go up and some of them are randomly shaken according the "Brown" value.

Step 10:

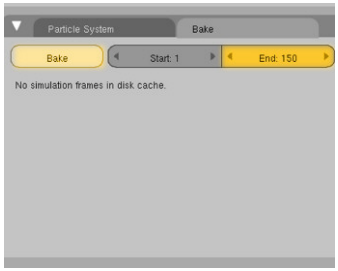
In the Particle System panel, set the "Amount" value to 3500, "End" to 150 and "Life" to 25. In "Emit From", check "Random" and "Even" and switch from "Jittered" to "Random".

Step 11:

In the Extras panel, access the "Neg" drop-down menu in the Vertex group section and choose the "Emivert" group that you created previously. Now, particles will only be emitted from vertices contained in this Vertex group!

by François Grassard

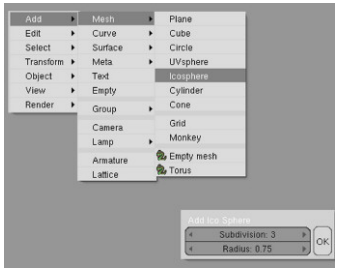
Step 12:



Enter the Bake panel, just next to the Particle System panel. Set the "End" parameter to 150 and press "Bake". Blender will create a new directory, next to the ".blend" file on your hard drive and fill it with a bunch of files (one per frame of animation). All of the cached data will be saved there.

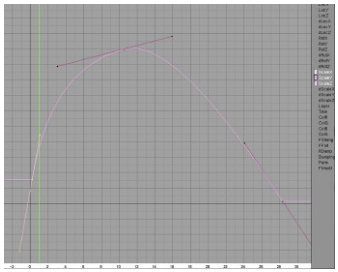
B) Generate a bunch of spheres:

Step 01:



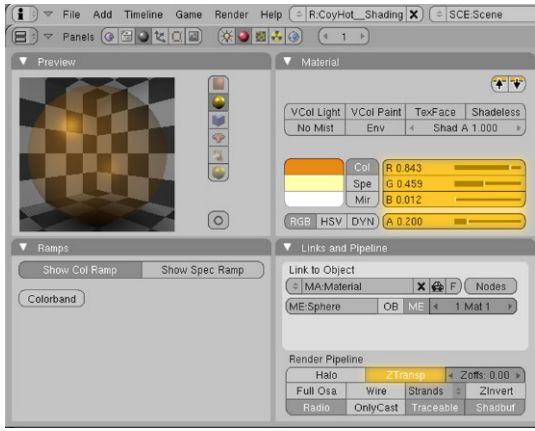
Create a new Icosphere and set the "Subdivision" parameter to 3, then press the Tab key to exit Edit mode and press [Alt+R], then [Alt+G] to reset their position and rotation parameters. Place this Icosphere at the center of the world coordinates.

Step 02:



Animate the size of the Icosphere over time. Create the first keyframe at frame 1 and set the Scale to something close to zero along the X, Y and Z axes. At frame 10, create a new keyframe and set the three scale values to 0.2. Finally, create the last keyframe at 30 and set the scale values to zero. Tweak the curve tangent (in the Ipo Curve Editor) to create a kind of paraboloid.

Step 03:



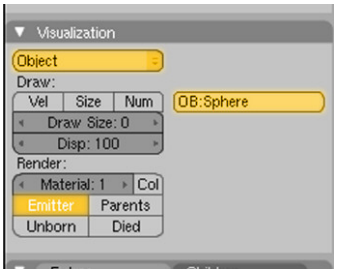
Select the Icosphere and add a new shader to it. In the Material panel (using on the color swatches), set the "Col" (color) value to a Hex code of D77503, and the "Spe" (specular) value to F7F69B. Set the "A" (alpha) value to 0.2 and check the "ZTransp" button (in the Links and Pipeline panel) to enable transparency without using Raytracing.

Step 04:

In the Shaders panel, set the "Spec" value to 2.0 and "Hard" to 200. Do not activate Raytracing (neither "Ray Mirror" nor "Ray Transp"), because the render time could be really long and the "Depth" values will never be high enough to get a good result.

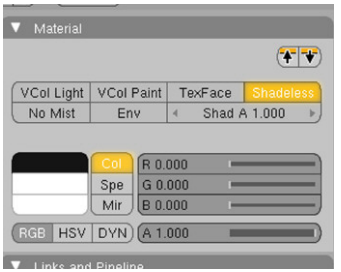
by François Grassard

Step 05:



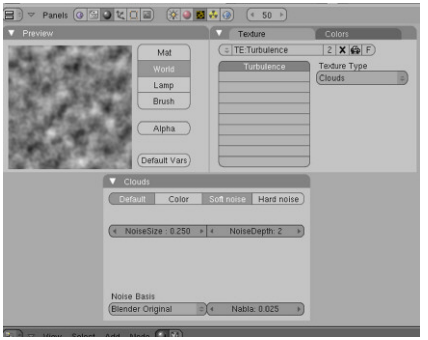
Select your particle emitter (Suzanne) and in your particle parameters, enter the Visualization panel. Click on "Point" and change it to "Object". In the "OB" field type "Sphere", referring to the animated Icosphere. Click on "Emitter", to render your Suzanne mesh in addition to the particles.

Step 06:



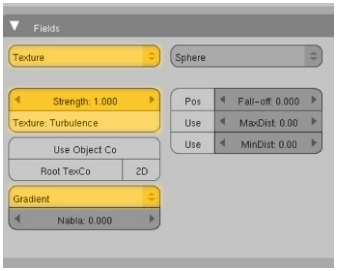
Select Suzanne and add a new shader to it, setting the diffuse color to black and checking the "Shadeless button". This will help to visualize the shape of the emitter as the flames come off of it.

Step 07:



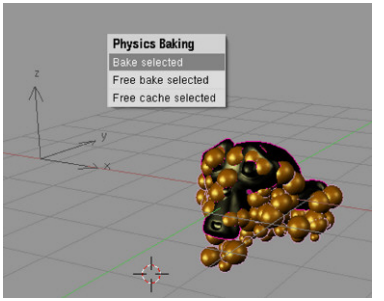
Press [F6] several times to display the Texture buttons, click on "World", then "Add New", and set the "Texture Type" to "Clouds". Leave all the parameters on the default settings and name this new texture "Turbulence".

Step 08:



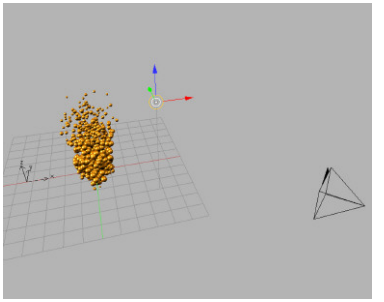
Add a new Empty to the scene (Spacebar›Add›Empty), select it and press [F7] several times to reach the Physics buttons. In the Fields panel, switch to "Texture" in the drop-down menu and enter "Turbulence" in the "Texture" field. Switch "RGB" to "Gradient" and set the "Strength" to 1.0.

Step 09:



Select Suzanne, press [Ctrl+B] and choose "Free bake selected" then [Ctrl+B] again and "Bake selected" to delete and create the cache again (since we've just added a new "Texture" field).

Step 10:

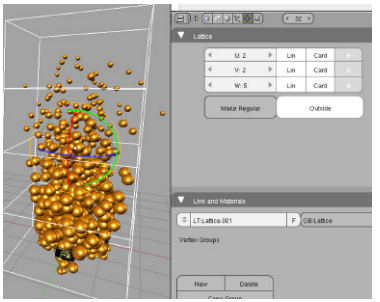


Add a Camera and Lamp to your scene and set the Lamp's energy to 3, then move it to in front of the upper-right area of your particle system, between the particles and camera.

by François Grassard

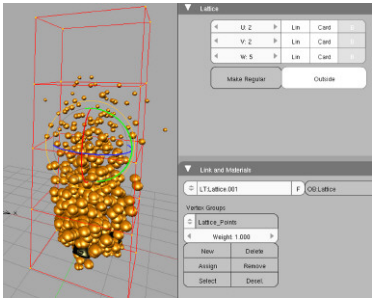
Step 11:

Add a new Lattice to your scene, and set it's "U" and "V" values to 2, and the "W" value to 5.



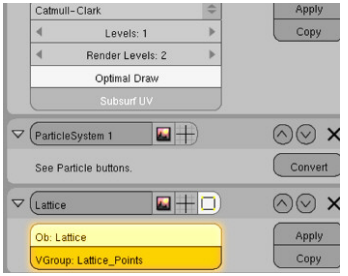
Step 12:

Use the Tab key to enter Edit mode and select all control points of the Lattice using the [A] key. Create a new Vertex Group called "Lattice_Points" and assign all the selected points to it.



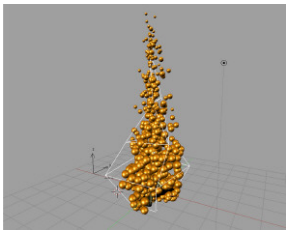
Step 13:

Exit Edit mode, select Suzanne and add a new Lattice modifier to it. Enter the name of your lattice in the "OB" field and enter "Lattice_Points" in the "VGroup" field, so as to distort only the particles and not the emitting mesh..



Step 14:

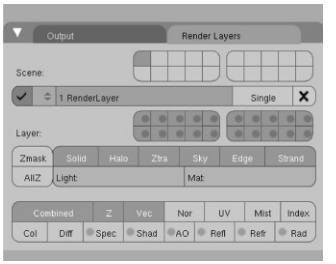
Select the lattice, switch to Edit mode and move the upper points to contract and twist the top of the flames. One it's done, leave Edit mode and press Play using [Alt+A] to check your animation.



C) Rendering and compositing:

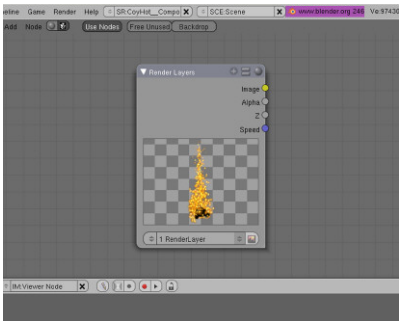
Step 01:

Now that the motion of your flame is set, we have to add the final magic touch to our render using compositing tools. Press [F10] in a Buttons window to show the render parameters. Go to the "Render Layers" panel and press the "Vec" button.

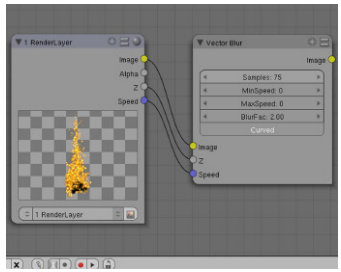


Step 02:

Open a Node Editor window, select "Composite Nodes" and "Use nodes". There should now be a RenderLayers node and Composite node. If not, add them using Add›Input›RenderLayers and Add›Output›Composite.

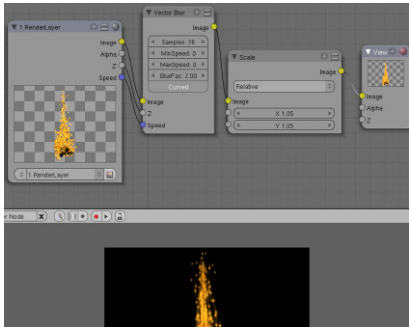


Step 03:



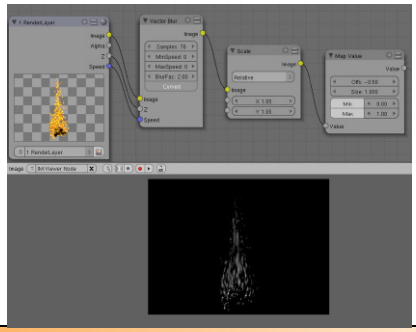
Now use Add›Filter›Vector Blur for the next node, and connect the "Image", "Z" and "Speed" outputs of the Render Layer node to the same inputs of the Vector Blur node. Press the "Do Composite" button in the Anim panel of the Scene buttons window, and launch a render using [F12] to see the current composite. With just this setup, we've turned a bunch of spheres into a decent flame.

Step 04:



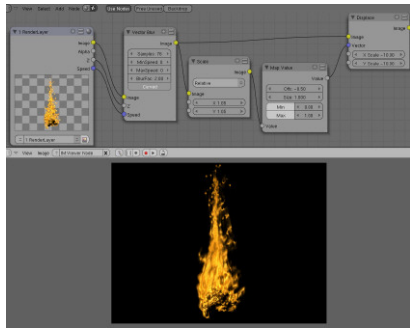
Next use Add›Distort›Scale and plug the "Image" output of the Vector Blur node into the same input of the Scale node, then set the "X" and "Y" values to 1.05.

Step 05:



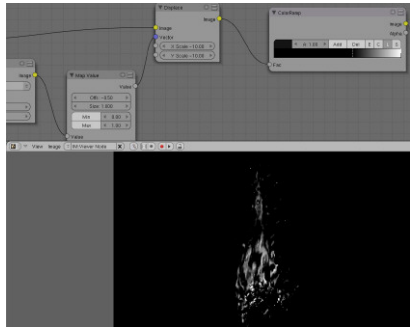
Add a Vector›Map Value node, set the "Offs" value to -0.50 and plug the "Image" output of the Scale node into the "Value" input of the Map Value node.

Step 06:



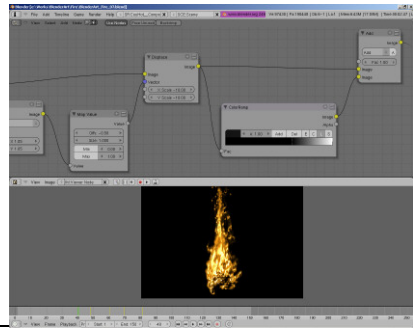
Add a Distort›Displace node and set the "X Scale" and "Y Scale" values to -10. Plug the output of the Map Value node into the "Vector" input of the Displace node; and now the "Image" output of the Vector Blur node into the "Image" input of the Displace node.

Step 07:



Add a Converter›ColorRamp node and move the black marker to the right, to about the 60% position on the gradient. Plug the output of the Displace node into the "Fac" input of the ColorRamp node.

Step 08:



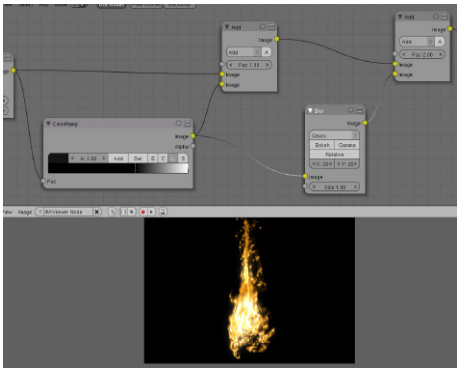
Add a Color›Mix node and switch from "Mix" to "Add" mode. Set the "Fac" value to 1.0, and plug the output of the Displace into the first "Image" input of the Add node and the output of the ColorRamp into the second "Image" input.

by François Grassard

by François Grassard

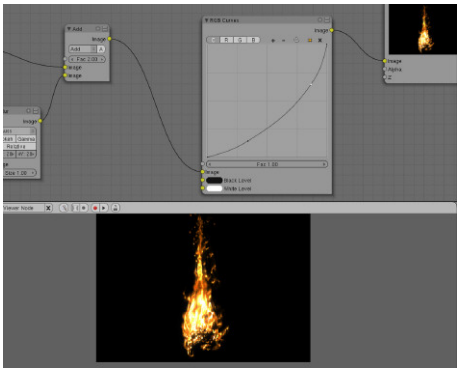
Step 09:

Duplicate the Add node using [Shift+D] and set the "Fac" of this duplicated node to 2.0. Add a Filter›Blur, node, change the mode from "Flat" to "Gauss", and then set the "X" and "Y" values to 20. Plug the "Image" output of the ColorRamp node into the "Image" input of the Blur node. Plug the output of the Blur node into the second "Image" input of our duplicated Add node. Finally, plug the output of the original Add node into the first "Image" input of the duplicated Add node.

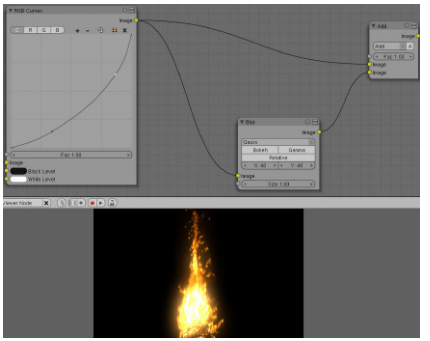


Step 10:

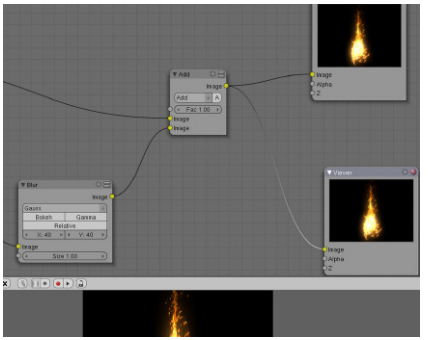
Add a Color›RGB Curves node, and plug the output of the Add node into the "Image" input of the RGB Curves node. Add two points to the "C" curve and change the profile to mimic an exponential curve.



Step 11:



Add another Filter›Blur node, and change the mode from "Flat" to "Gauss", and the "X" and "Y" values to 40. Add a new Color›Mix node, switch it to "Add" mode and set the "Fac" to 1.0. Plug the RGB Curves output into the "Image" input of the new Blur node and the first "Image" input of the new Add node. Finally, plug the output of the Blur to the second Image input of the Add node.



Step 12:

Lastly, add an Output›Viewer node and connect the output of the last Add node to the "Image" inputs of the Viewer node just created, and the original Composite node.

Step 13:

In a Buttons window, press [F10] to get back into the render parameters. Setup the desired output options (file format, compression and location) for your video launch a render of all the frames of your animation by pressing the "Anim" button or [Ctrl+F12].

There's no smoke without fire:

Using the new particle system in Blender 2.46 (especially the "Reactor" feature, which can generate a new particle system at the death of another one), you can combine this fire tutorial with one that shows how to simulate realistic smoke.

n the near future Blender will reintegrate a wonderful feature (previously integrated, but deprecated during the production of the Peach Project) that makes possible the editing of all particle types, and not just hair/fur. With this feature, you can sculpt the shape of your fire without using a lattice, as we've done here.

The future of the particle system is really promising, and Blender does not have to be ashamed of its capabilities when compared to other software.

by François Grassard

by Daniel Salazar aka ZanQdo

Introduction

This scene required generating the movement of thousands of objects in a very specific way, the basic idea was to build the gazebo out of small parts falling from the sky.

Since animating all off these objects by hand was clearly not the best and most flexible idea, we decided to do a python script that would handle the motion of each piece based on the final desired position.

The final motion script can be seen here.

The basic Math

My approach consists of two simple and predictable algorithms, one that controls the fall (Z axis) and one that controls the translation (X and Y axis) of the object.

For the Z movement I've used the free fall formula that we all learned in school: (Wikipedia and Google are great tools for people with bad memory/math skills like me)

*z = (- 0.5 * Gravity * Time ** 2) + (Initial_Velocity * Time) + Initial_Altitude*

You can see the relevant function in line 87 of the script. So using a free fall formula gave me nice, believable Z Loc values to start with; now for the translation part a simple 2D linear extrapolation formula was used, it works like this:

a = 2

b = 10

First find out the range between a and b with a subtraction

b - a = 8

Now divide the range in some large value, lets say 100

8 / 100 = 0.08

This gives us 100 equally spaced steps between positions a and b. For example if we want to find out the middle position we do

*0.08 * 50 = 4*

And offsetting the value of a

4 + a = 6

The same can be done for any other value between 0 - 100:

*(b - a) / 100 * 85 + a = 8.8*

The cool thing about this is that we can use values greater than 100 to get extrapolation, lets say using a value of 175:

*(b - a) / 100 * 175 + a = 16*

In the script I'm using this to get the X Loc and Y Loc values of each object out of their final desired position (b) and the center of the gazebo (a) which I get from an empty object that I can even animate with oscillating movements to get interesting effects, check lines 68 and 78 for final implementation.

by Daniel Salazar

Adding Variation:

Python comes with a module called 'random' which is great for easily adding variation to the results of each run of the formulas, its actually not really random but pseudo random, which means it takes a given seed number that you can specify and it'll generate many numbers out of that, and if you later give it the same seed and ask for the same thing, it will generate the exact same numbers. This characteristic is great because it allows us to keep things fully predictable as I mentioned before.

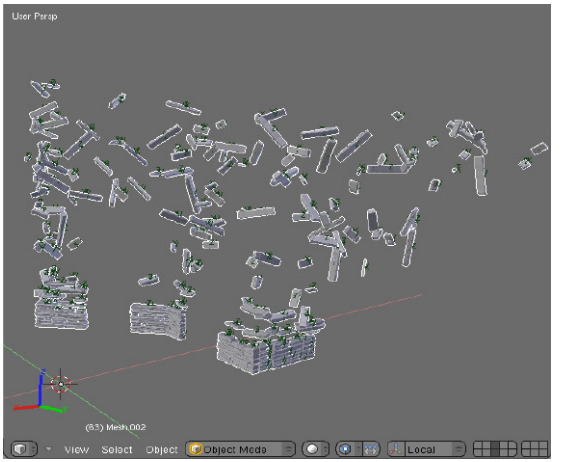
If you go back to the Loc and Rot functions you will see how random generated numbers are used to give variation to the results. Line 39 shows how to set a different random seed for each object that will stay the same in time, avoiding jittering in the object's movement.

Making it fit

Prior to running the main script, I've saved all object's rest positions into good old Game-Engine properties using this one time use script. It also saves a starting time offset value for each object. The example I'm sharing here was used for the roof; it sets the start time according to a distance from center formula as you can see in line 38 and that's why the roof's tiles start falling into position from the outer part to the center. I'm also using a little bit of random generators here.

So this script uses lots of random numbers and different formulas and we still need to control the final or rest position of every object so that the whole bunch correctly builds the gazebo; this is when the predictable formulas and pseudo-random generators save the day.

The trick is to predict were the object is gonna be at the end of the movement and to offset the entire motion accordingly so that when it actually reaches the rest position you have it right were you wanted it. In lines 96-98 and 113-115 I'm calling the location and rotation formulas but I'm feeding them the time value of 100 which marks the end of the movement (like seeing into the future!) Then in lines 130-135 I simply subtract this value to the results of the real time formulas and I have all objects magically landing right in the center of the world. Finally I add the location values previously saved in each object with the secondary script and now they land in the desired position!



Smoothing the movement:

For now the formulas used, especially the extrapolated translation one, return a linear movement which produce a very mechanical motion. However it feels nicer to have a "slow out" or deceleration right before each piece reaches its rest position. This is what IPO curves do best so I've used one to control all formula's time range. Check line 52 for the function that gets the current value of a curve simply named "IPO", and if you check the returns of all the rotation and moving functions you'll see how this value is used there to remap the otherwise linear time range.

Now that you understand how it works you can try it your self with these instructions:

- 1- Add game engine properties to the objects, storing their rest location and also a start time offset; this is done with the second script that works on selected objects. The Start property is then set by different formulas that you can write for any specific case.
- 2- Add your objects to a group named "Main", when ever you want to disable their animation just remove the objects from this group.
- 3- Add the main script as a ScriptLink triggering on "FrameChange" and scrub trough the frames!
- 4- You need a "Time" curve in an Empty object on an IPO called simply "IPO" going from (0, 0) to (100, 10); this controls the interpolation of each piece so you can add "slow out" effects, etc
- 5- You need another Empty object named "Center" in the center of the scene or in the center of the build effect, you can animate this object to achieve nice oscillating movements
- 6- Run animation with Alt A or timeline

Credits:

Wow Factor produced by martestudio.com

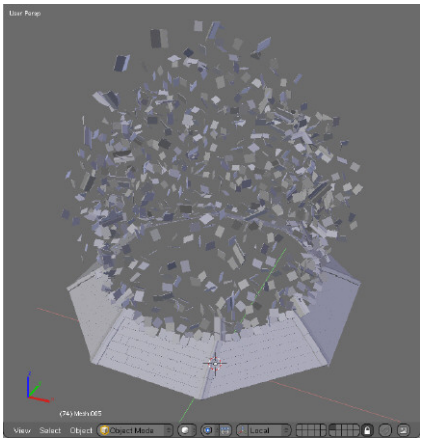
Render by Oliver 'imshadi' Zúñiga

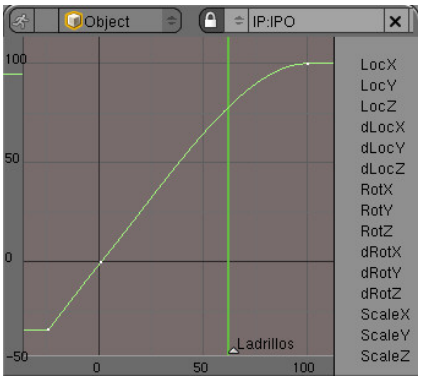
Special thanks to Joshua 'aligorith' Leung and Geoffrey 'briggs' Bantle for developing tools necessary to finish this shot.

contact@zanqdo.com

You can watch the final shot here:

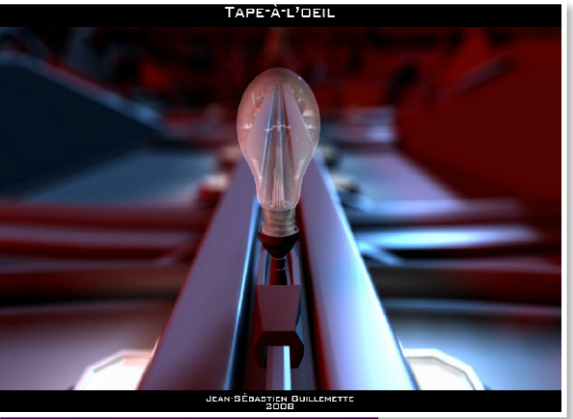
http://zanqdo.com/tmp/Rancho.mov (Please Do not Direct Link).





by Daniel Salazar

MAKING OF: 'Meet the eye a.k.a Project LightBulb'



by Jean-Sébastien Guillemette

Introduction

This scene required generating the When Sandra Gilbert contacted me back in April to write an article for the BlenderArt magazine about my short movie "Tape-à-l'oeil", my first reaction was one of happiness and surprise. I immediately accepted thinking it would be great to go back at my work and give an insight to others on the production of it. After some thought, it became obvious that I didn't use any special tricks, or hidden features in Blender.

My short movie was definitely minimalist in many ways, including in the tools I used to create it. I thought some more, and I came up with an idea for the article. Why would I make an article on the tools and tricks I used, which are probably already explained in at least a dozen other tutorials, while instead I could write an article on the experience I gained from this project. One thing I learnt for sure is that constraints greatly help imagination. In this article, I will try to dig into my production, and how I worked to get the job done in time.

First of all, you guys should watch the short movie if you didn't already. It can be viewed on our newly opened website. If you're asking yourself right now "isn't that the website of Jonathan Williamson, a.k.a Mr_bomb on Blenderartists.org?", make sure to read the blog to understand everything. For the others, go read as well! Once on the website, just go to the gallery, and into the animation section.

Now that you watched, I'll explain to you in which context the short movie was created. The short was created as a thesis project for my 2 years of cinema study in college. The subject given to us was "The cinema and myself", and yes, I was the only one coming up with an animated movie.

And this was exactly the source of my main problem. There were just three rules to follow: the movie had to be between 3 and 5 minutes long, shoot the movie in a maximum of two days, and only use the editing rooms for a maximum of 25 hours. Fortunately for me, my teacher exempted me from the last two.

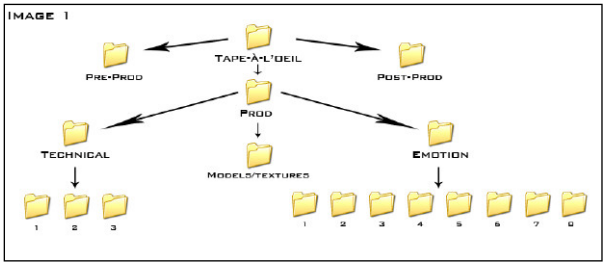
This left me with the whole two months of February and March and 8 days into April to model, texture, animate, render, have the music composed, and edit the whole thing together. As everything that can go wrong does go wrong, I received a call on February 1 from a research team asking me to create a two minute animation.

To make a long story short, I accepted, and it took me a full month to finish it. I officially started working on my short movie on March 3rd. By now, you're probably asking yourself why I'm explaining all this stuff. Let me answer that these side-stories made my short what it is, since my short movie basically became my life for that period of time. So, now, my main problem should be clear: How to create 5 minutes worth of animation in such a small time period.

First thing I did, was to strip down my scenario. I removed some shots which were only there because they were cool, which led me to refine my actual intentions with this piece, and later on made me change the title to "Tape-à-l'oeil" which translates to something like "eye-candy". Then I started to split my scenario into specific "scenes" and sections. Since I'm really not good at -

by Jean-Sébastien Guillemette

drawing, and didn't have much time to waste, I didn't draw any storyboards [but you definitely should if you plan on making a short movie], but I was familiar enough with my scenario to know exactly how I was going to represent each sequence. I started by creating a series of folders representing each scene. You can check the image 1 to see my folder tree.

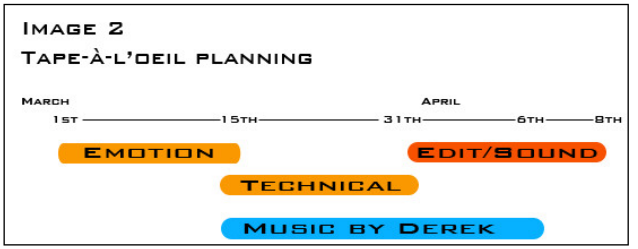


My "Pre-prod" folder was filled with my scenario(s), but also with inspirational pieces. I always keep an inspiration and reference folder for when I am in doubt as to what feeling I want to give to a certain shot. May it be songs, images or texts, I kept everything that inspired me for this short movie in there. In my "Prod" folder, I kept every file that I created during the production.

First, I created a folder for the models and textures. All the models that I used from scene to scene, a light bulb and a character in this case, were kept in that folder so I can append them at any time and link them from scene to scene. The textures were minimal, if not nonexistent, in this short, so that part of the folder stayed mostly empty. I then created two main folders for the production.]

I wanted to do it this way for several reasons. First off, my short is about the duality between the technical side of cinema and animation, and how some things that need to be done to create a movie are far from being artistic, and the emotion and creativity that emerge from these "technical constraints". I wanted my production to reflect that, and I wanted to work on my project as if the two distinct parts of my movie were two separated projects that were blended (pun intended) together. I also did this to get a scale, and a feeling at how big the "emotion" part was for the sole reason that I needed to create this one first because I needed some music for it. I had to finish that part at least 2 weeks before the deadline so my music composer, Derek McTavish Mounce, could have time to create a masterpiece, which he did (thank you Derek).

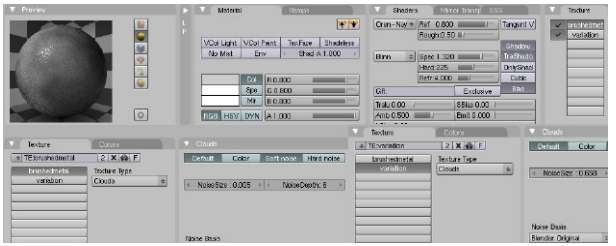
Talking of deadlines, I created a small schedule so I could compare my progress with the actual time passing by. Check out Image 2 to see how I organized my time. How wrong I was... I ended up finishing the emotion part around March 28, sent it in a hurry to Derek, then worked my way through on the technical part of the short in an intense two days and finished rendering the whole thing by April 4th thanks to my new quad core computer.



Derek definitely worked even more intensively and ended up giving me the song at 3h am on Sunday the 6th of April. I then proceeded to school, where I had to do the final edit, with all my rendered files and the fabulous song from Derek, and finished editing everything by Monday afternoon, more than 24 hours before the deadline. So, that is basically how the overall project went. Now let's get some CGI insight.

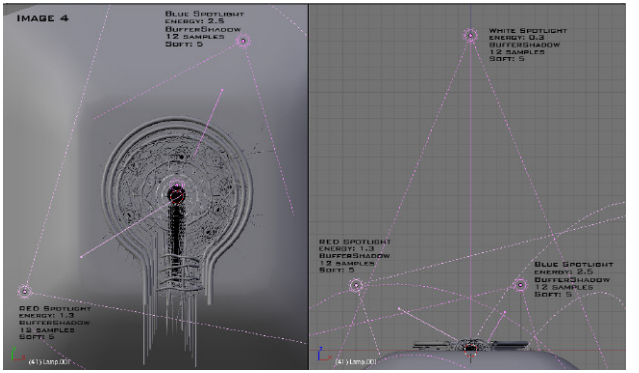
by Jean-Sébastien Guillemette

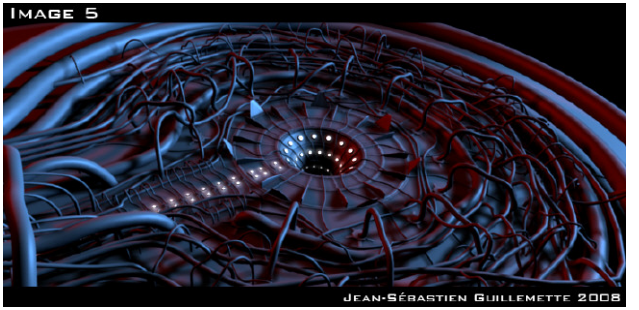
One thing I knew since the beginning, was that rendering the whole thing, even on my quad core, would take quite a lot of time. So I decided to go minimalist. I used no textures at all, except a few procedurals for the light bulb's metallic part. Check the link at the end of the article to download the light bulb .blend and check Image 3 to see my material setup.



The "brushedmetal" texture was then applied with the "nor" option to create a bumpy surface on the object. The use of very simple procedural textures and materials greatly improved the rendering time during the production and helped me finish on time. I went in the same direction with the modeling and lighting. I was keeping the very minimum for the shot to look good [or decent at least]. Here's an example of the "most complex" lighting I had in the short. It is during the transition sequence between the first and the second part, and it's when the light bulb is inserted in some kind of projector, from which the character emerges.

Since this particular scene needed to clash with the precedent, more technical base scenes, I needed to push the emotion out of the lamp. That is why I used powerful coloured lights, red and blue, so the colours could oppose themselves as the technicality, and the creativity are opposing themselves in my short. Since using Ambient Occlusion to fake global illumination was practically out of question due to the long render time it would demand, I played with a few different lighting tricks to simulate it, and ended up using a single lamp (which ain't really a trick...) to soften the shadows of the two other, very powerful, lamps. Note how the two coloured lamps are very low from the ground perspective. This was specifically done to create long shadows that would blend with the shadows created by the other lamps. This kind of low angle lighting really adds to the emotions and drama of a scene, just as a sunrise or a sunset does. Check Image 5 for a render of the scene. As for the smaller lights in the hole and along the light bulb path, I simply used a shadeless white material on the object, and then added a "shadowless" sphere lamp above it.





by Jean-Sébastien Guillemette

Another scene that was quite important was the "light bulb field" scene, in which the character flies through about three thousand light bulbs, and then shapes them into a circle (Image 6 & 7). To generate the field, I used Alan Dennis' Cloud generator. I simply used it to generate randomly placed vertices over a quite large area. I then appended my light bulb model, stripped it down from it's major details (filament inside the bulb, removed the subsurface modifier, removed the refraction and reflection from the material, etc..) and dupliverted it (Object menu (F7)), on my generated points cloud. Then comes the funny node part.

To light these light bulbs up, I simply duplicated the cloud mesh, and created a new "halo" object which consisted of a single vertex with an orange-ish halo material and dupliverted it on the duplicated cloud like I did with the light bulb. I then placed this new duplivert system on another layer. Check Image 8 for the node system of this scene.

IMAGE 6

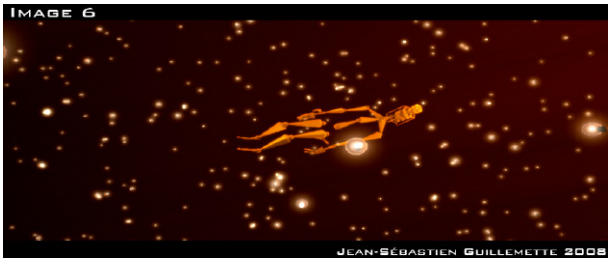
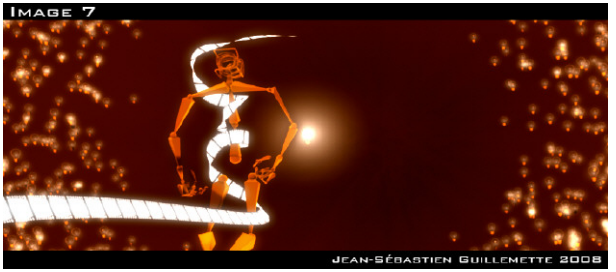


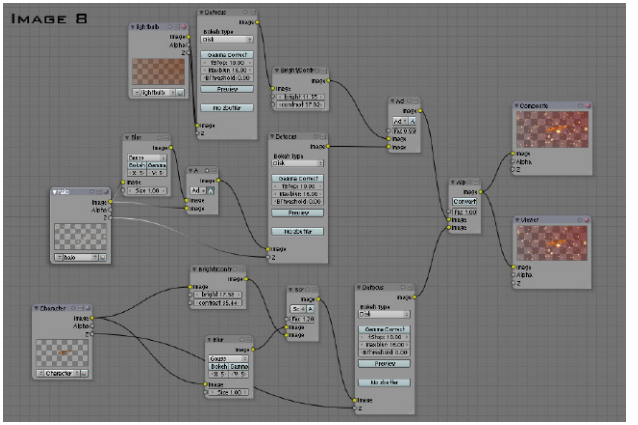
IMAGE 7



As you can see, I separated the whole scene in three distinct render layers. First off, there is the light bulb field render layer (At the top of the node system). On it, I use a Defocus filter to generate my Depth of field effect (which I use throughout the whole movie), and also a Brightness/Contrast node to add more contrast to my scene (another quick trick instead of using a more complex light setup).

I then "add" the halo render layer on top of the light bulb layer. To add a bit more bloom to these halos, I blur them up and then "add" the blurred halo on top of the rendered halos. And I use the exact same trick on the character to give it slightly better "wow effect" (Decided to do that since a basic armature setup ain't that cool to look at).

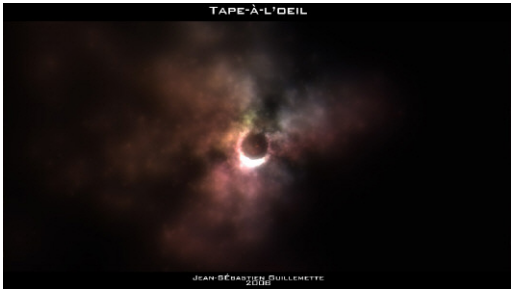
IMAGE 8



After that, when came the time to animate all these light bulbs and shape them into a circle, I used a cheap cinema trick. During a cut in-between two shots, I created a new .blend which contained all the same objects, placed exactly like they were in the previous .blend. Instead of using only duplivert though, I also used a particle system.

by Jean-Sébastien Guillemette

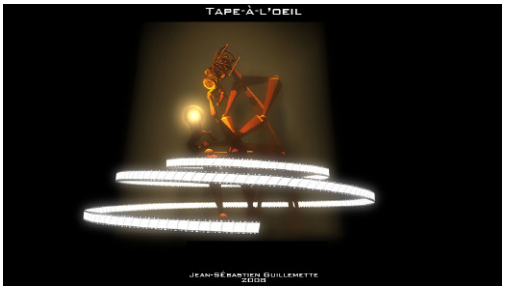
Since each object making up the clouds (there were 3 sections forming the whole cloud) had 1870 vertices, I added a particle system to them which would emit 1870 particles. Then I clicked the "from: Verts" button in the particle system menu (and made sure the "face" button was unselected). I then



changed the particle start and end value to 0 and 1 respectively. Doing this makes the object emit all the 1870 particles at the same time before the animation actually starts on frame 1. I also made sure there was no speed whatsoever, so all the particles would stay in place once they were emitted. Then, I used a simple empty with a "Vortex field" propriety and voilà, my particles were slowly forming into a circle. Since I didn't want to have them expand forever, I simply created a new .blend with a fractally subdivided cylinder that had about two thousand vertices (at this point, the new shot didn't have to match perfectly what was in the previous shot).

Conclusion

As a conclusion, I'd like to thanks Jonathan Williamson for his help, critiques and for the motivation he gave me throughout the couple weeks I worked on this. Make sure you pass by our website over at http://www.montagestudio.org to view my short movie, but also the work of Jonathan. If you have any specific question about the project, or request for specific



files, I could always make available what you ask for. This project was created with an open mind, and begs the spectator to stay open minded throughout the whole 5 minutes of it, so I don't really have any reason to not "open" the content of it. The only reason at the moment keeping me from releasing all the files is the time it would take to actually organise the folders and files properly with text explaining each of them. Releasing so many files won't do any good if taken out of context, so I'm not willing to do it if I don't have the time to do it right. As I said, if you have any specific request, email me at jsguillemette@hotmail.com or jsguillemette@montagestudio.org.

Here is the lightbulb object:

http://www.montagestudio.org/Ecks/ProjectLightBulb/lightbulb.blend

And this is the character during the very final scene,

http://www.montagestudio.org/Ecks/ProjectLightBulb/Character_LePenseur_jsguillemette.blend

I hope you enjoyed this article as much as I enjoyed doing this project and writing on it.

- Jean-Sébastien Guillemette

Introduction

Since the implementation of fluid simulation in Blender, I have been fascinated by the realism produced – it just looks right. In our common experience, we expect that a liquid will always move under the influence of all forces acting upon it, including gravity, until it finally reaches a stable state, if possible. A very curious thing about a waterfall, for example, is that the shape of the bend (due mostly to momentum and gravity) is static, while the flow is dynamic.

Fluid Art: the idea.

What I had not seen among the various fluid simulation demonstrations online was something like a water fountain, where water would flow upward and fall back down, revealing a shape that water, otherwise at rest, would not have. A sophisticated fountain, I thought, would be intriguing with its increased complexity. A simple implementation of this in Blender would have:

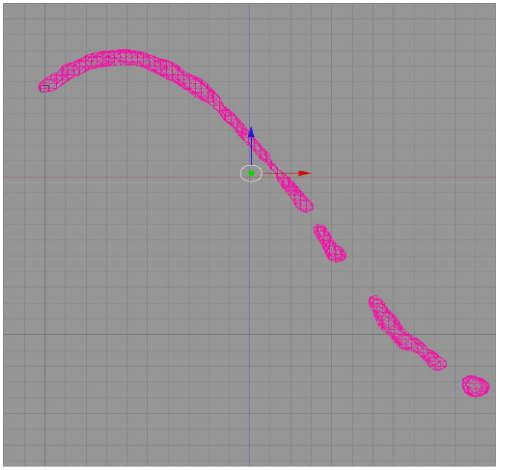
Several inflow objects

- Each one with the same vertical inflow velocity (Z)
- Each one with a specific horizontal inflow velocity (X and Y) so that the total effect is interesting

I decided to use sixteen inflow objects, arranged in a circle, as viewed from above, and direct the stream from each one up and toward the center. At the center, then, all streams would collide and stop moving in the horizontal direction, so that only vertical motion (down) remains. If you visualize this, it is like a tree with a trunk and branches.

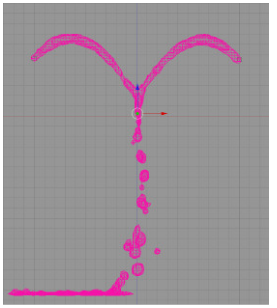
Starting point: how does a simulated fluid stream behave?

At this point, what was needed was some specific information about the distance and the shape of the stream for a given fluid velocity. I did many experiments with one stream to find simple values of geometry and velocity. Here is a screenshot of satisfactory results, as viewed from the side. (insert Figoo2.png) The inflow object was a cube of small size to make a narrow stream, having a Z-velocity of 1 (up) and an X-velocity of 1 (positive, to the right), and Y-velocity of 0. It is important to note two things: (1) the stream crosses the origin of the grid, whereas later, the origin will become the center of the circle of inflow sources; (2) the final domain resolution of the mesh needs to be as high as possible; I was able to achieve 200.



by Jack Harris

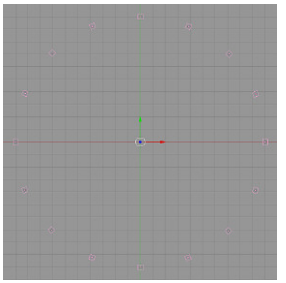
Two streams collide: fluid becomes an obstacle as well.



To create a second stream, I made a mirror copy of the first stream across the origin, did the simulation and checked the result for a reasonable intersection of the two streams at the center. The two streams merged quite nicely and flowed downward at the center. The second inflow object had the same Z-velocity, but with an X-velocity of minus 1 (negative, to the left), and Y-velocity of 0.

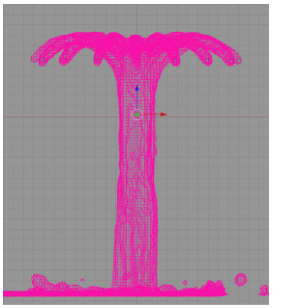
Branching out: do the math.

The final configuration of inflow sources was a circle of sixteen cubes, equally spaced along the circumference, centered at the origin, as viewed from above. Each inflow object required calculation of the components of velocity in X and Y, which I have not included here, except to point out that the magnitude of the total horizontal velocity was equal to 1, and the direction was toward the center. The result of baking this setup yielded the desired tree shape.



Add some color: fake material is not easy.

Like elaborate water fountains on display at night, the lights are part of the art. I added colored spotlights above at strategic locations to give green reflections on the branches, and area lights along the trunk for brown reflections. The difficulty with this method was confining each color to specific parts of the tree, with no overlap. To take full advantage of this lighting setup, I changed the default material settings. Under the Shader tab, I set the amount of reflection at 1, the degree of specularity at 2, and the hardness at 1.



Conclusion: and a challenge.

I rendered several seconds of this simulation. Amazing. It was obvious that the effect worked. This project has been difficult and, at the same time, fun. What satisfies my curiosity is the good results from Blender's fluid simulation function. And it is fascinating to watch a flowing liquid form a familiar, or at least, an interesting shape.

My challenge to any reader is to try doing this with different geometries, velocities, lights, materials, even to use IPO curves to vary the fluid velocities with time. And have fun.

Jack Harris

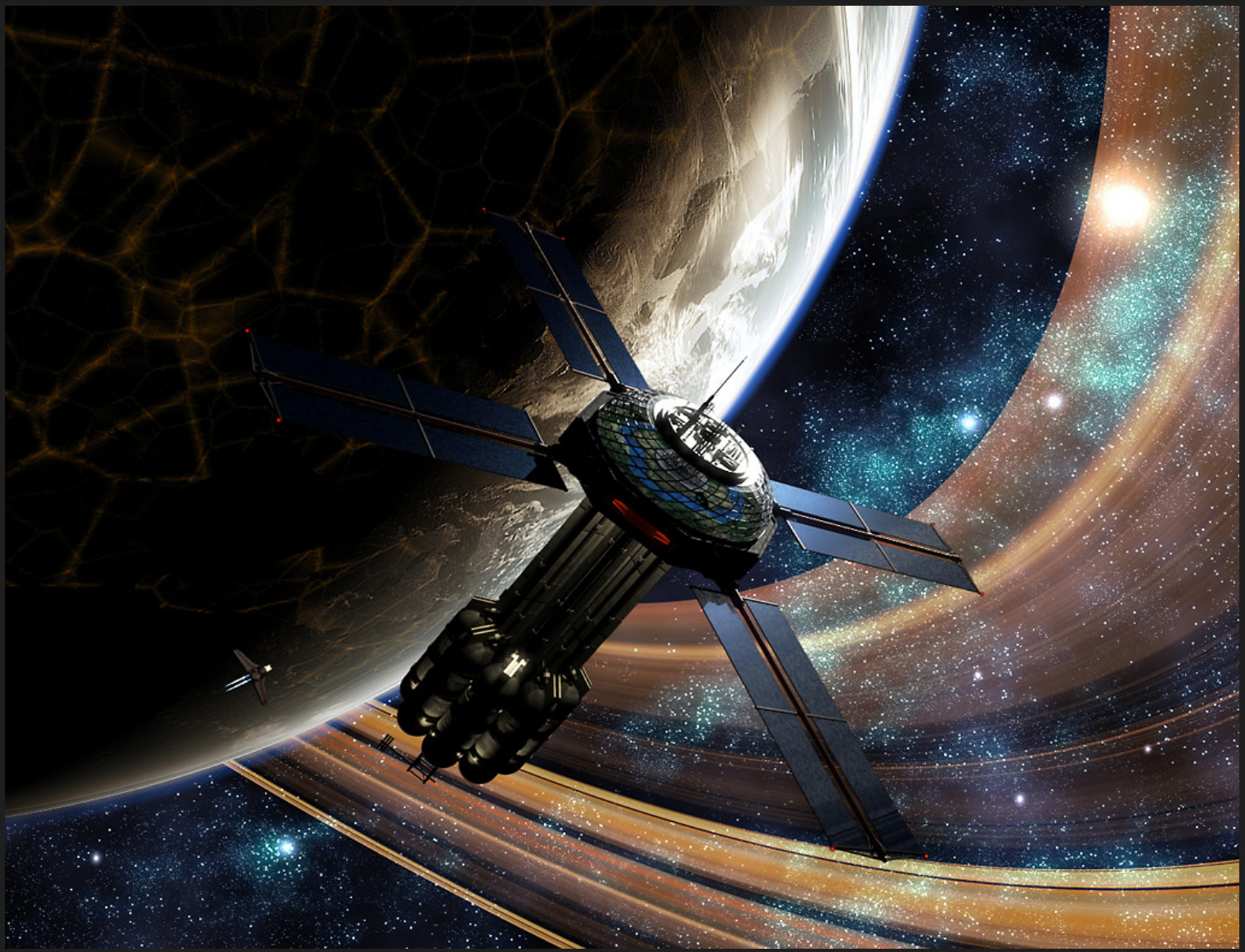
Jack Harris (okchoir) is a retired electrical engineer of northern Texas, USA, and has been using Blender since 2004. He enjoys choir singing and reading, and is currently building a steel-framed, retirement home.

by Jack Harris

Christian Storay - Lucy

Elfak - The Alchemist

Kevin Whitfield - Station



Kevin Whitfield - Space Scape

Jeepster - Statue

Captif, dit le Brandebourg - Jeepster

Igor Kekeljevic - Godiva Project

Want to write for BlenderArt Magazine?

Here is how!

1. We accept the following:

- Tutorials explaining new Blender features, 3dconcepts, techniques or articles based on current theme of the magazine.
- Reports on useful Blender events throughout the world.
- Cartoons related to blender world.

2. Send submissions to sandra@blenderart.org. Send us a notification on what you want to write and we can follow up from there. (Some guidelines you must follow)

- Images are preferred in PNG but good quality JPG can also do. Images should be separate from the text document.
- Make sure that screenshots are clear and readable and the renders should be at least 800px, but not more than 1600px at maximum.
- Sequential naming of images like, image 001.png... etc.
- Text should be in either ODT, DOC, TXT or HTML.
- Archive them using 7zip or RAR or less preferably zip.

3. Please include the following in your email:

- Name: This can be your full name or blenderartist avtar.
- Photograph: As PNG and maximum width of 256Px. (Only if submitting the article for the first time)
- About yourself: Max 25 words .
- Website: (optional)

Note: All the approved submissions can be placed in the final issue or subsequent issue if deemed fit. All submissions will be cropped/modified if necessary. For more details see the blenderart website.

Upcoming Issue 'Theme'

Issue 17

Lights! Camera! Action!

- Lighting: Lighting tips and tricks
- Rendering: Using render layers
- Compositing: Rendering for final output
- Video Editing: Getting it all together
- Sequence Editor
- NLA

Disclaimer